Project Proposal ANLY 515

Risk assessment, analysis, modelling, and visualization of returns for IT, Pharma and Energy sectors.

Sectors	Stock 1	Stock 2
IT	Microsoft MSFT	Cisco CSCO
Energy	Exxon XON	Chevron CVX
Pharmaceutical	Amgen AMGN	Johnson & Johnson JNJ

Submitted by:
Gaurav Gade

Date: 04 August 2020

Table of Contents

Introduction

This project proposal aims to conduct a comparative study of tools and techniques for performing risk assessment, analysis, modelling and calculation of returns for a portfolio consisting of two stocks each from IT, Pharma and Energy sectors for the timeline from July 1 2010 to July 1 2020.

Objective

The objective of the project is to develop a robust model that can predict the stock performance and market risk for a portfolio using statistical tools and methods.

Project scope

For quantitative and qualitative analysis, this study will be focussed on applying selective statistical methods on a portfolio of 6 stocks over a period of 10 years.

Portfolio Description

The portfolio considered for this project consists of 3 industrial sectors - Information Technology, Pharmaceuticals and Energy. For each sector, two stocks have been chosen as follows.

- Information Technology:
  - ❏ Microsoft (MSFT)
  - ❏ Cisco (CSCO)
- Pharmaceuticals:
  - ❏ Amgen (AMGN)
  - ❏ Johnson & Johnson (JNJ)
- Energy:
  - ❏ Exxon Mobil (XOM)
  - ❏ Chevron (CVX)

Portfolio Timeline

The timeline considered for this portfolio is from July 1 2010 to July 1 2020.

Findings

The findings will include the actual results from running the statistical analysis by sector and portfolio.

Conclusions

The conclusions of the detailed project will be focussed on the following parameters:

- Stock performance
- Portfolio comparison from the 3 sectors
- Effectiveness of chosen methods in stock performance assessment
- Alternative methods for further analysis
- Challenges encountered and resolutions
- Future scope and learning outcomes
- Practical application of the developed model

Project techniques

For analysis, the approach involved importing the financial data for the above 6 stocks from yahoo finance. The next steps involved preparation of the datasets for analysis by masking out of scope data, cleaning up the datasets to ensure there are no null values and outliers. Further, we worked on grouping the data into monthly and weekly data. Next steps focussed on comparison of capital gains and calculation of returns. We used the following techniques for the stock risk modelling and analysis:

- Trend Analysis based on Closing prices
- Trend for closing prices with respect to market capitalization and volume
- Normalized Return value by Sector
- Plotting simple moving average
- Bollinger Bands Plot
- Comparing Capital Gains of Multiple Securities Over Time
- Logarithmic returns will be used for cumulative returns
- Stock performance analysis:
- Portfolio Return:
- Constructing Benchmark Portfolio Returns
- Value - Weighted portfolio
- Risk Assessment and Hyperbolic Distributions:
- Mathematical matrix calculations:
- Construction of Variance-Covariance Matrix
- Calculate the portfolio risk
- Suitable distributions for return
- Optimize portfolio based on Quadratic programming.

Project Plan:

Based on the techniques discussed above, our project plan followed the methodology given below. In the final report, we covered all the 6 stocks covering 3 sectors. In the project plan, we have shared the findings based on a few of the stocks and sectors of the final problem.

Trend Analysis based on Closing prices

Model Description

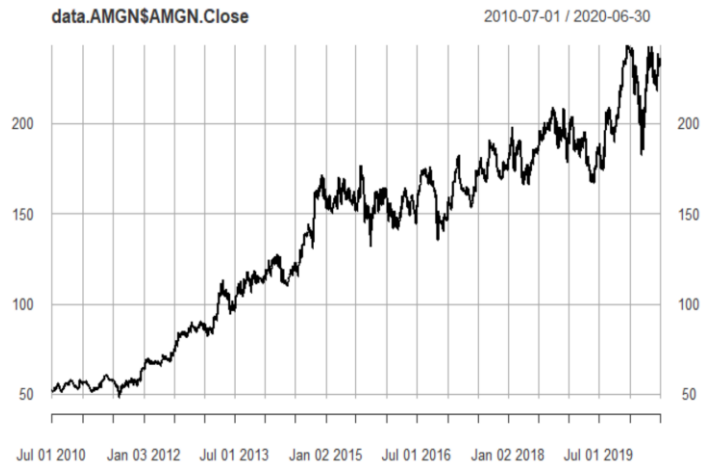
In this method, we have converted the data into a data-frame and calculated normalized value for each security. Then we have created an index for each security with values that equal the price of the security on each day divided by the price on July 1st., 2010.

We studied the closing prices for the 6 stocks in the portfolio and presented the results at a sector level. The closing price plot illustrates the overall performance of the stock during the period chosen for this study.

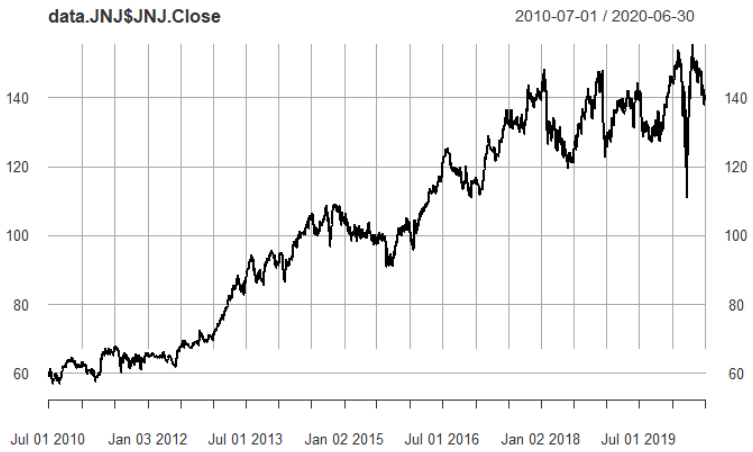
Output Analysis:

We have used a line graph to plot the securities.

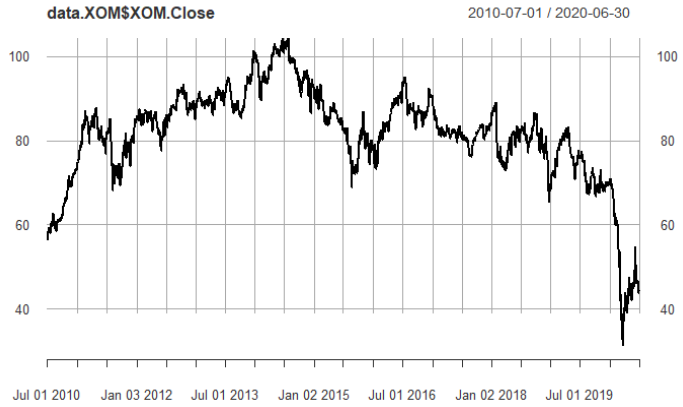
Amgen's closing prices



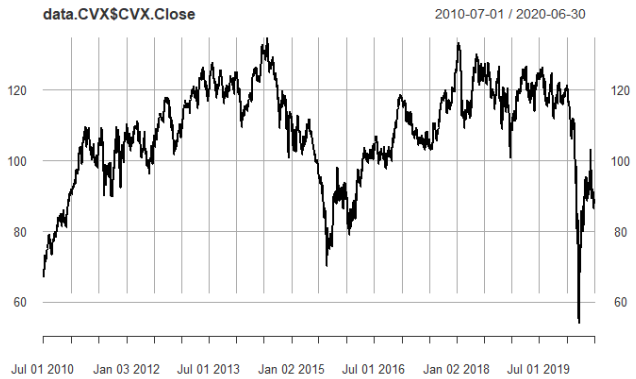
Johnson & Johnson's closing prices



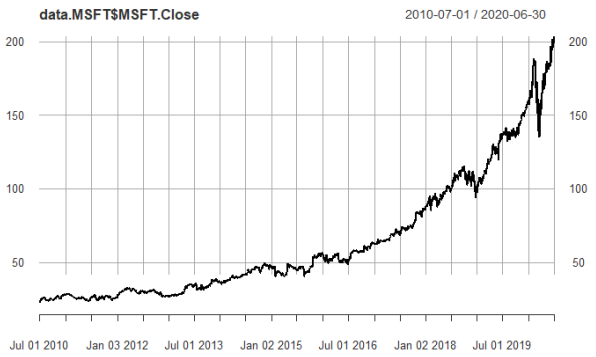
Exxon Mobil closing prices



Chevron closing price



Microsoft closing price



Cisco closing price

Conclusion/Recommendation:

By studying the plots of all stocks and sectors, we observed that the IT sector has performed significantly well driven by Microsoft. The energy sectors have not performed that well. We evaluated that there are no significant deviations or abnormalities indicating any major restructuring.

Trend for closing prices with respect to market capitalization and volume.

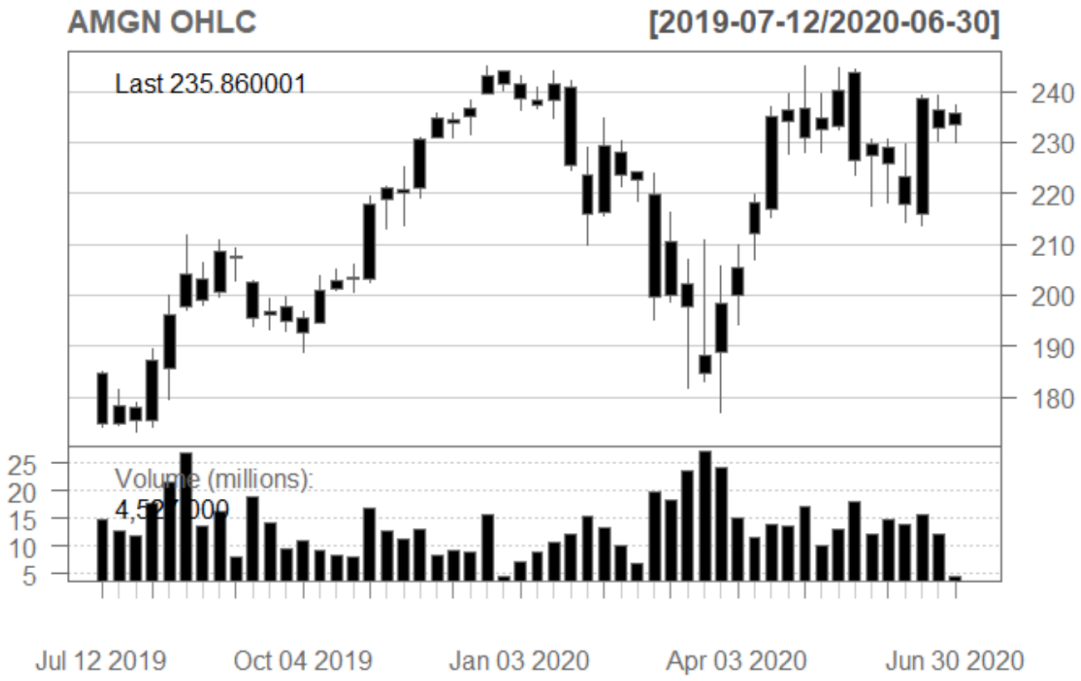
Model Description:

By using candlestick charts, we studied the adjusted closing price trends for each of our sectors. This helped us to understand which sectors have performed well over time. For example, we have built candlestick charts to compare stocks in the same sector (Pharmaceutical).

To build this candlestick chart we created a Open High Low Close (OHLC) object and used the chart series function to build the candlestick charts. We converted the xts object to weekly and monthly time series.

AMGEN and Johnson & Johnson comparison:

Using this tool, we were able to compare the stock time series and the volume (millions).



Output Analysis:

The first part of the above plots for Cisco and Amgen shows the trend of closing price. The lower part of the plot shows the capitalization over time. Following tools from the first two lectures were used for the purpose of this study:

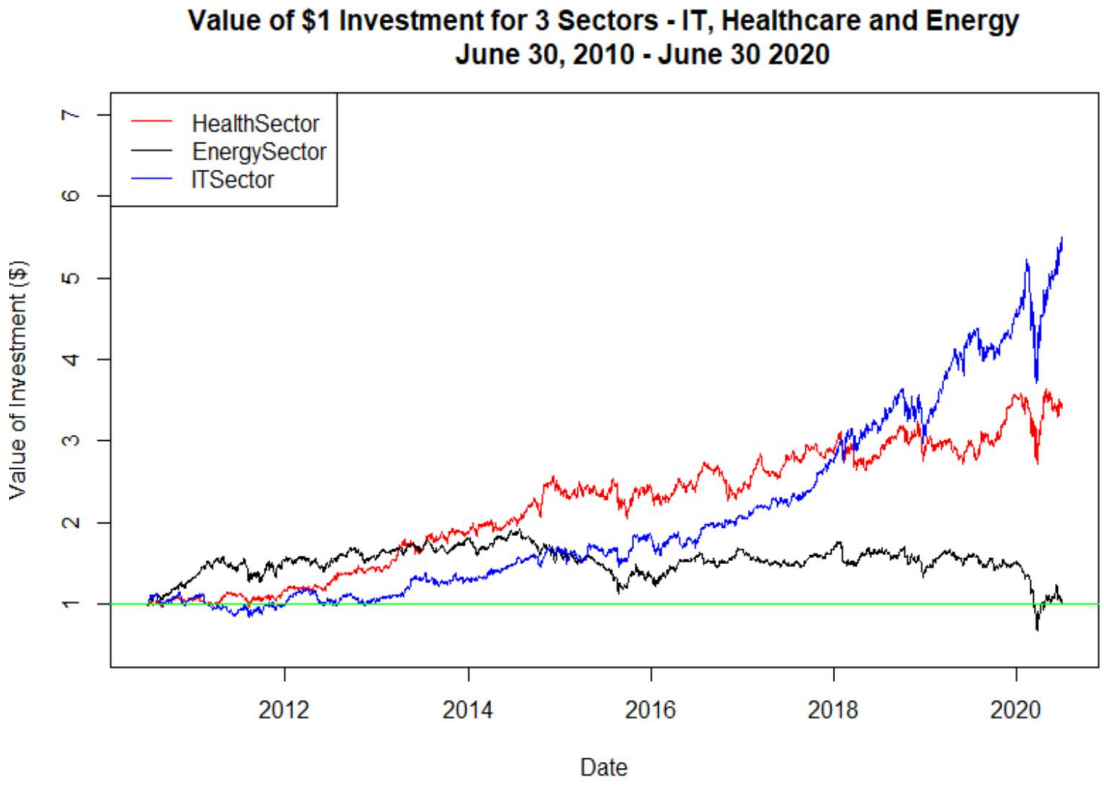
- Candlestick chart:
- Sector based analysis
- Sector wise comparison of returns from 2010 to 2020

Normalized Return value by Sector:

Model description

In the first part of the project, we used techniques to compare and assess individual securities. We used normalization to enable us to combine stocks from the same sector and present a sector-wise performance of returns over a period of time.

As we can see in the below graph, the value of a 1$ investment over time for 3 sectors IT, Healthcare and Energy can be easily compared.



We first converted the data into a data frame. Then we used the cbind function to bind the stocks in a single object. Then we calculated the normalized values of each security by creating an index for each security with values that equal the price of the stock on each day divided by the stocks' price on Dec 31st. 2010. Capital gains were calculated using the same normalized values.

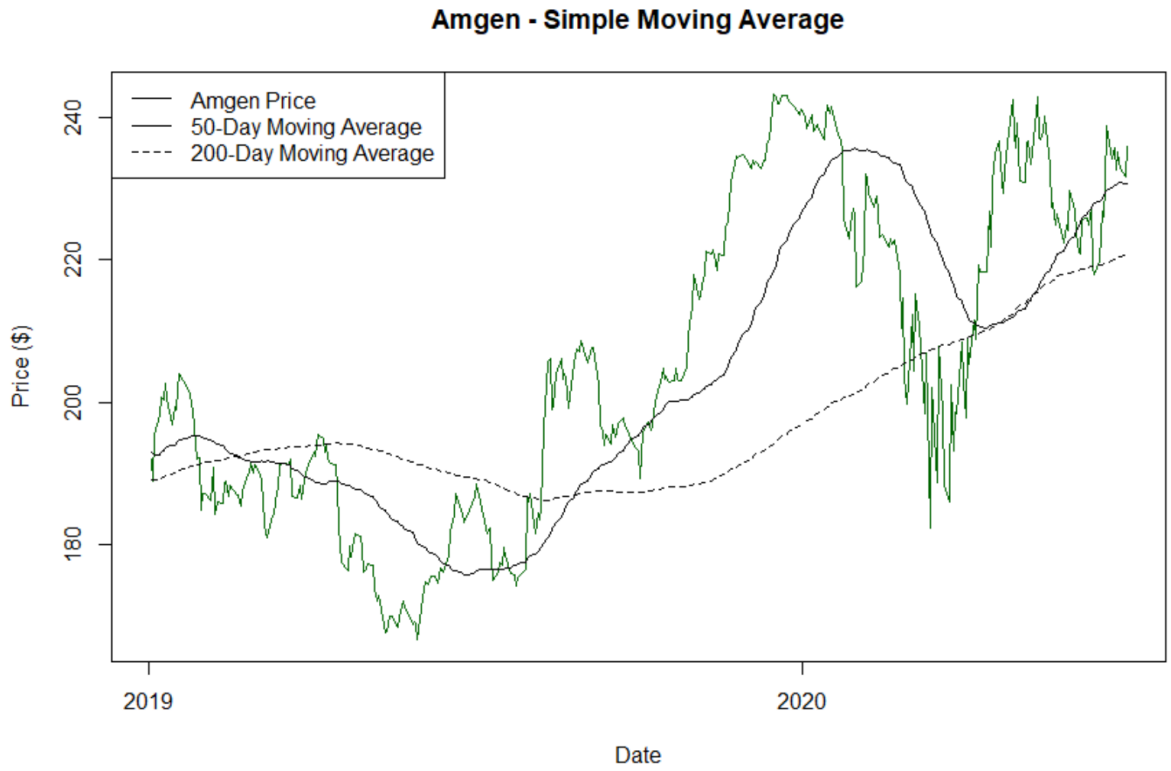
Plotting simple moving average

Model Description:

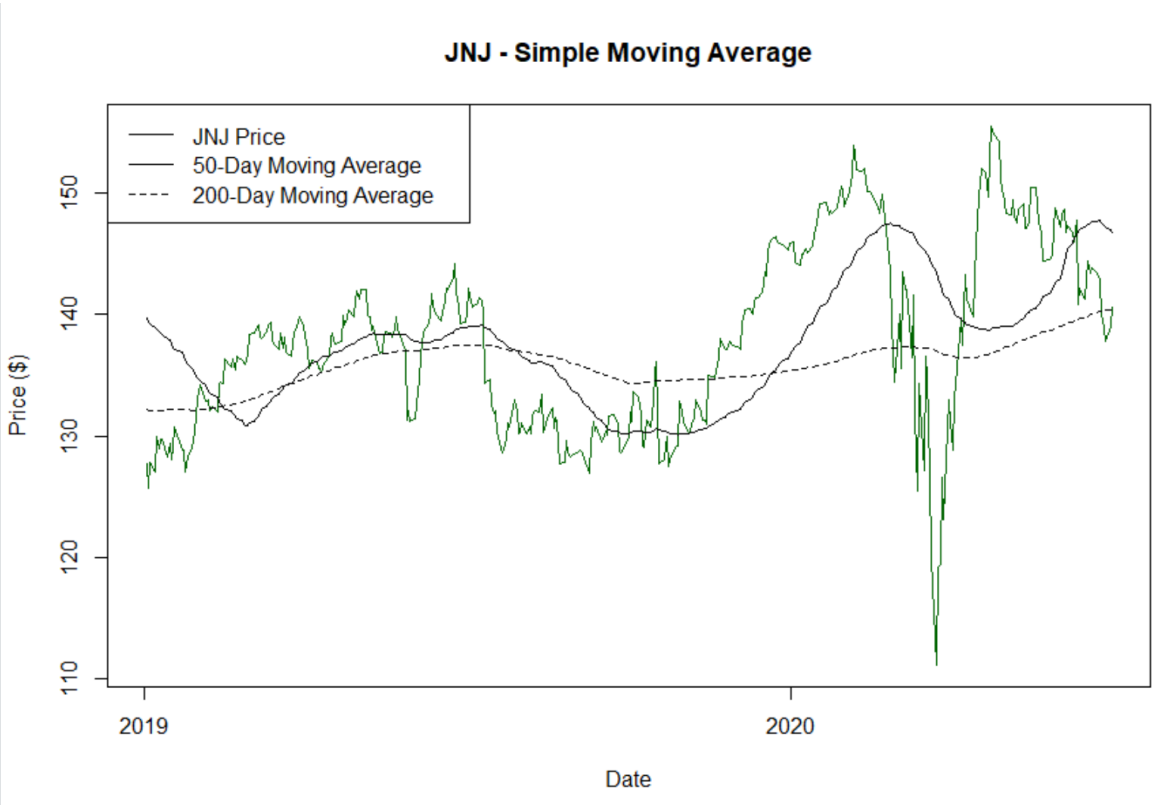
Using one of the sectors we first calculated the Rolling 50-Day and 200-Day Average Price. Then we used the merge function to combine individual stocks into their sector. Then we calculated the range of the y-axis following which we plotted the sma. This enabled us to compare the trends for Simple Moving Average Crossover for multiple stocks.

Simple Moving Average Plot for 2019- 2020

Output Analysis:

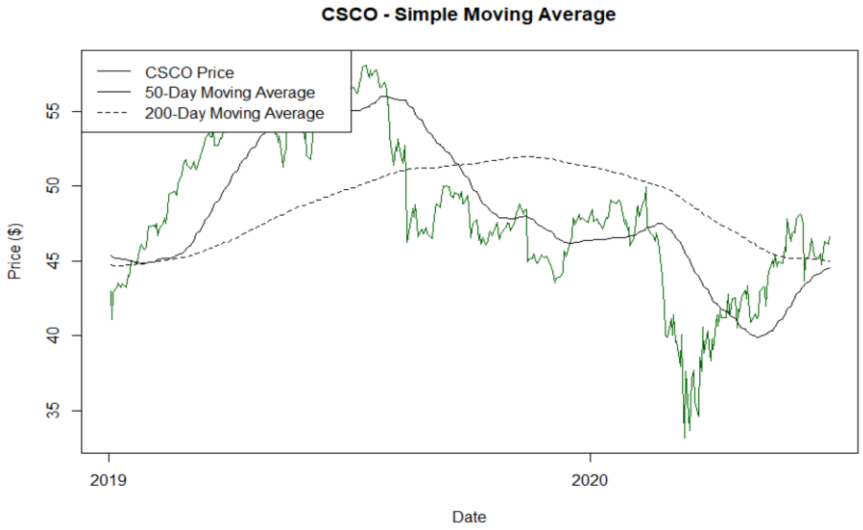


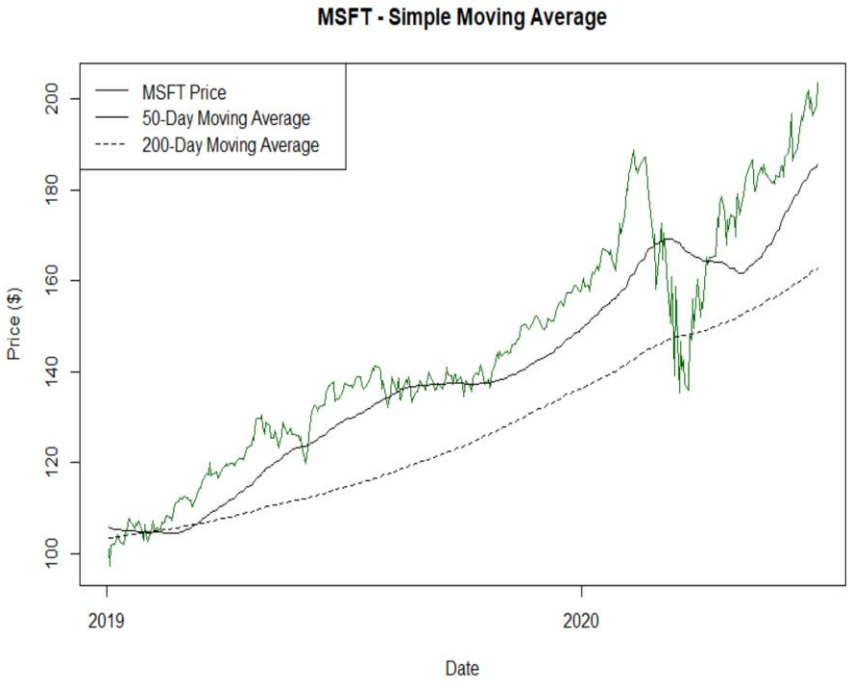
Amgen was around $240 at the start of Jan 2020 and it crashed to around $180 in March 2020. That corresponds to a drop of approx. 25% reduction in stock. The stock rebounded in April and is currently back to its starting price in Jan.



Conclusion/Recommendation:

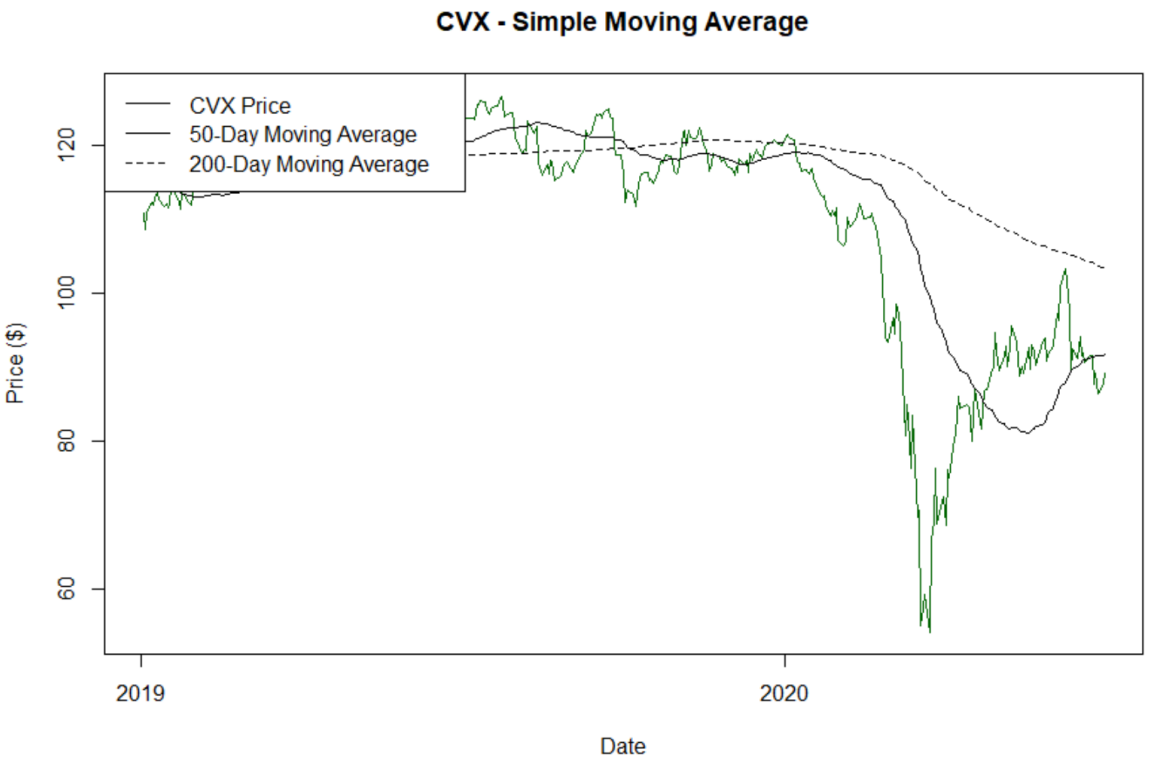
From the simple moving average graph plotted above, it seems like the pharma sector - Amgen and JNJ have similar trends over the period of 1 year from 2019 - 2020. There is a significant drop around March 2020 for both JNJ and slightly lesser than that for Amgen.

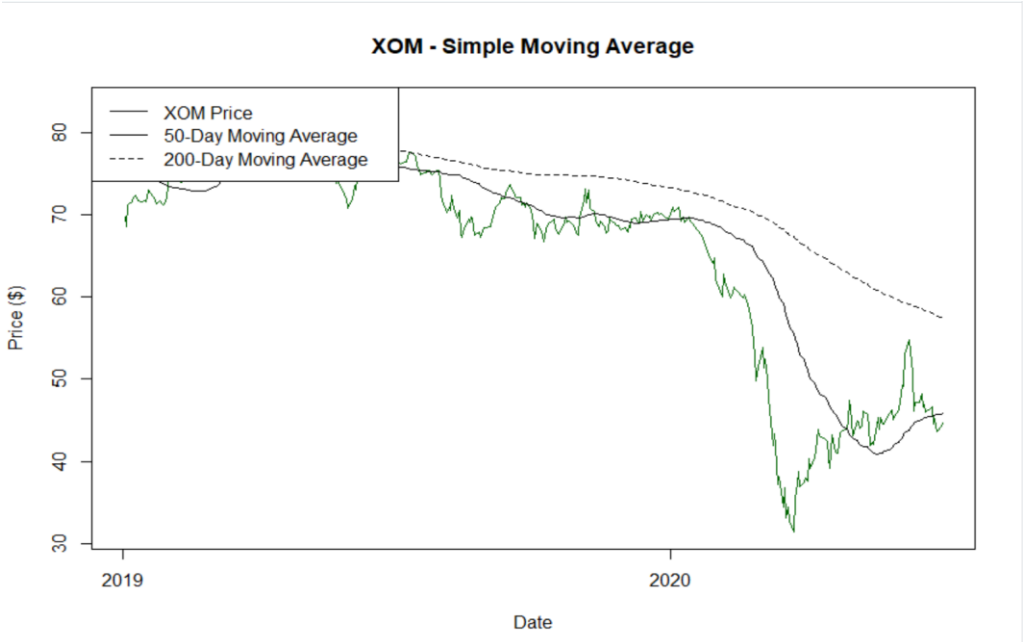




Conclusion/recommendation:

From the above SMA graphs, it seems like Cisco and MSFT's financials plunged around March 2020, it is more pronounced in case of Cisco and lesser compared than that in case of MSFT. Although over the 1-year period from 2019 - 2020, both the stocks seem to have increased in value, but Microsoft performed much better than Cisco and has approximately doubled their stock value.

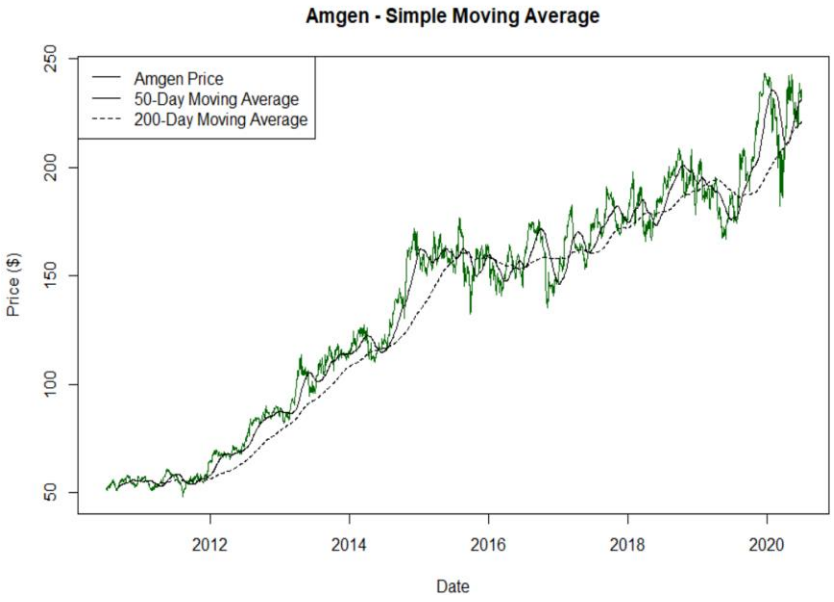


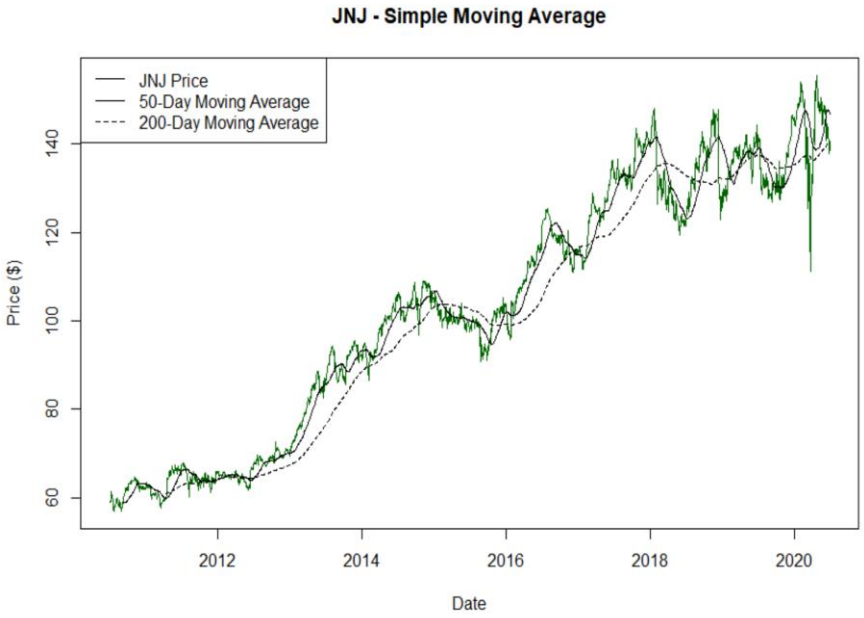


Conclusion/recommendation:

From the above SMA graphs, it is evident that both CVX and COM are heavily impacted due to the COVID -19 crisis. The above graphs are plotted with respect to the 1-year time period from 2019 to 2020. This is expected due to the lack of industrial activity and travel restrictions. It is expected that when things get better then we should see a spike in prices for both CVX and XOM and from the above graphs, we might safely conclude that buying these stocks will ensure huge profits for the buyer in the long term.

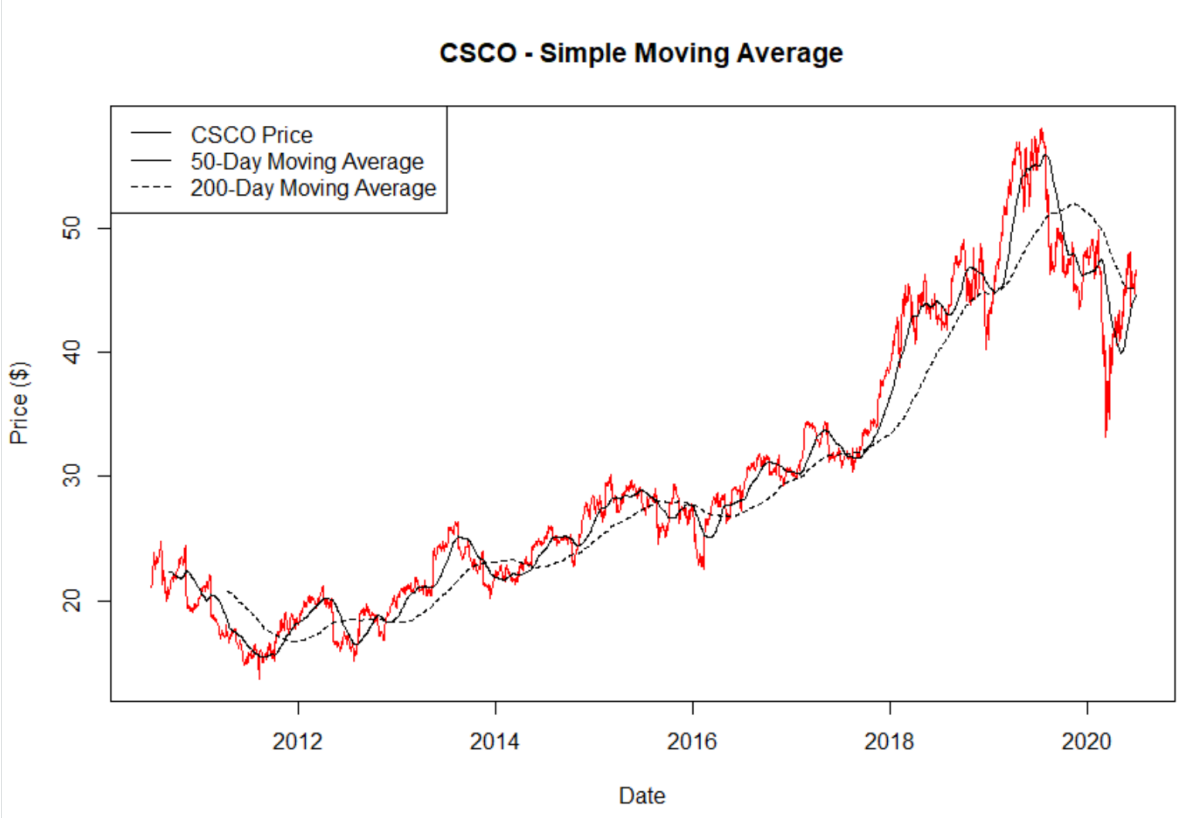
Simple Moving Average Plot for 2010 - 2020

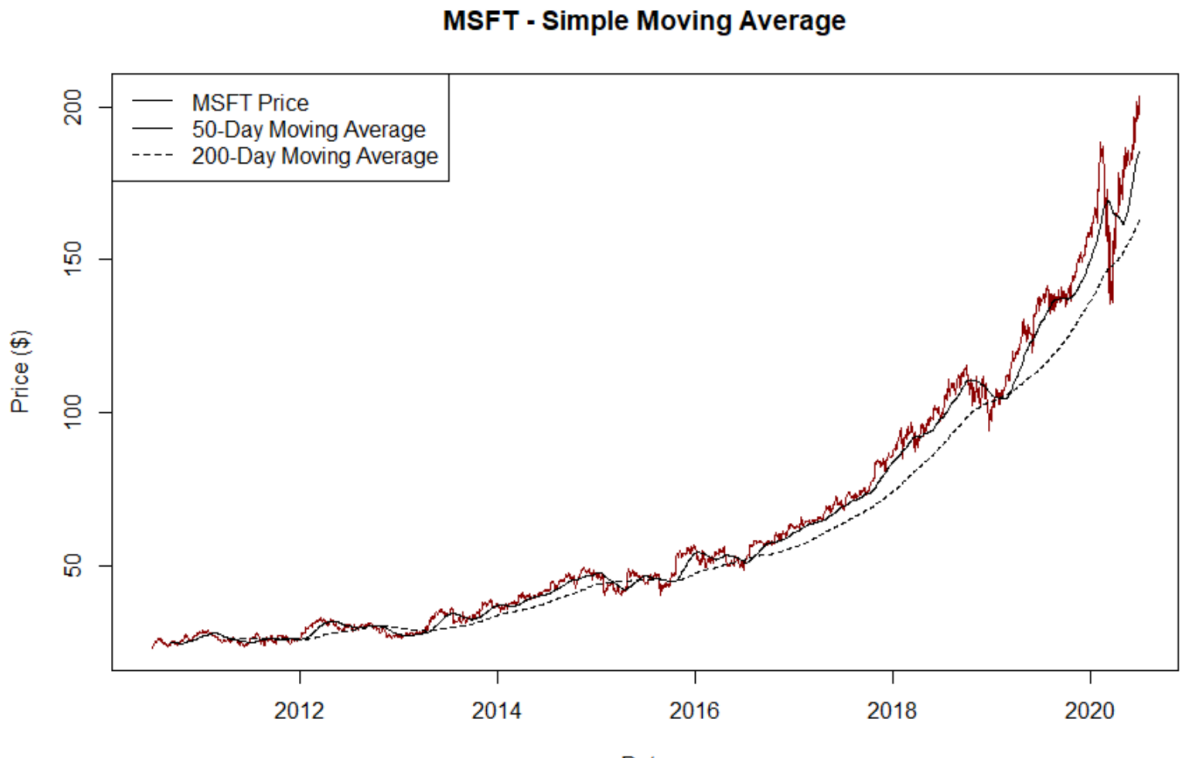


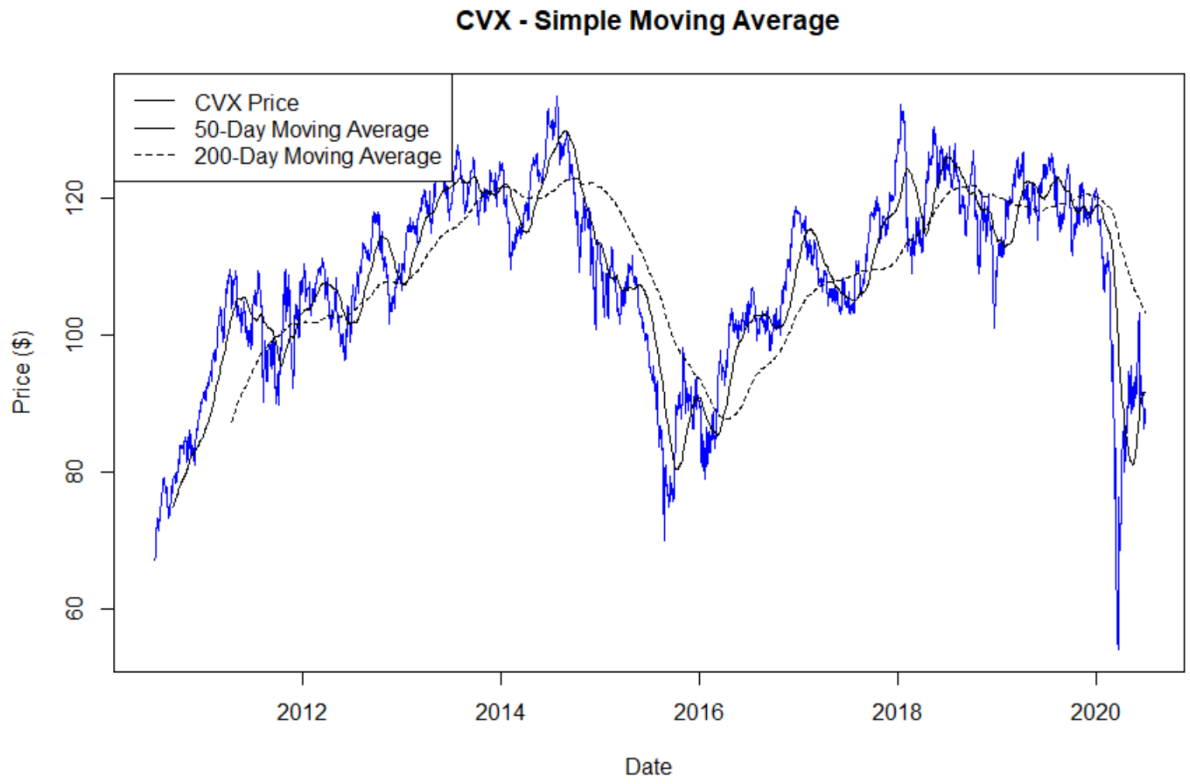


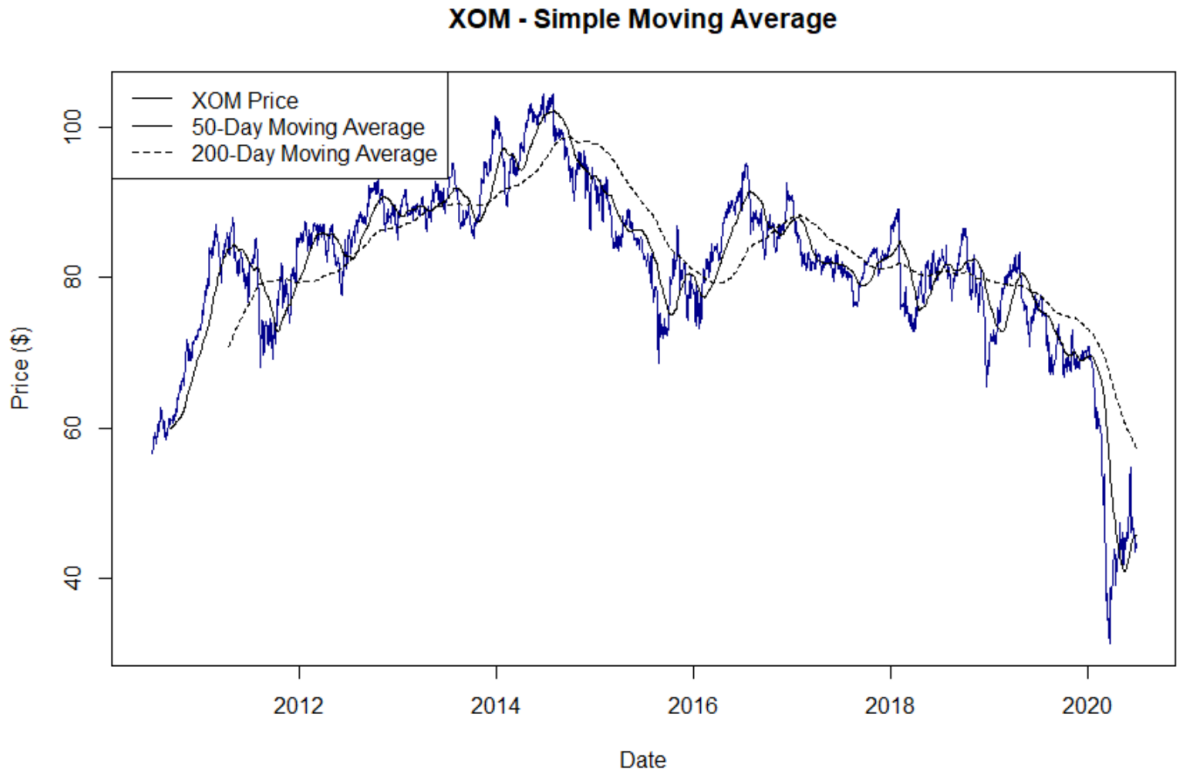
Conclusion/recommendation:

From the above graphs of AMGN and JNJ simple moving averages over the period of 10 years from 2010 - 2020, it is clear that both the stocks have done well and have appreciated significantly. AMGN stocks started around $50 in 2010 and looks like it is around $225 in 2020, which is almost 350% gain in 10 years. In the case of JNJ, the stock started at $60 and is currently around $150, which is approximately 150% gain in 10 years.




MSFT - Simple Moving Average
MSFT Price
50-Day Moving Average
200-Day Moving Average
Price ($)
50
100
150
200
2012
2014
2016
2018
2020


CVX - Simple Moving Average
CVX Price
50-Day Moving Average
200-Day Moving Average
Price ($)
60
80
100
120
2012
2014
2016
2018
2020
Date



Conclusion/recommendation:

In the case of the Technology sector, MSFT did exceedingly well over the period of 10 years. MSFT went from $23 to $200, which is a 700% increase. Cisco went from a start price of $25 to $45, over 10 years. So it looks like Cisco is not yielding so much.

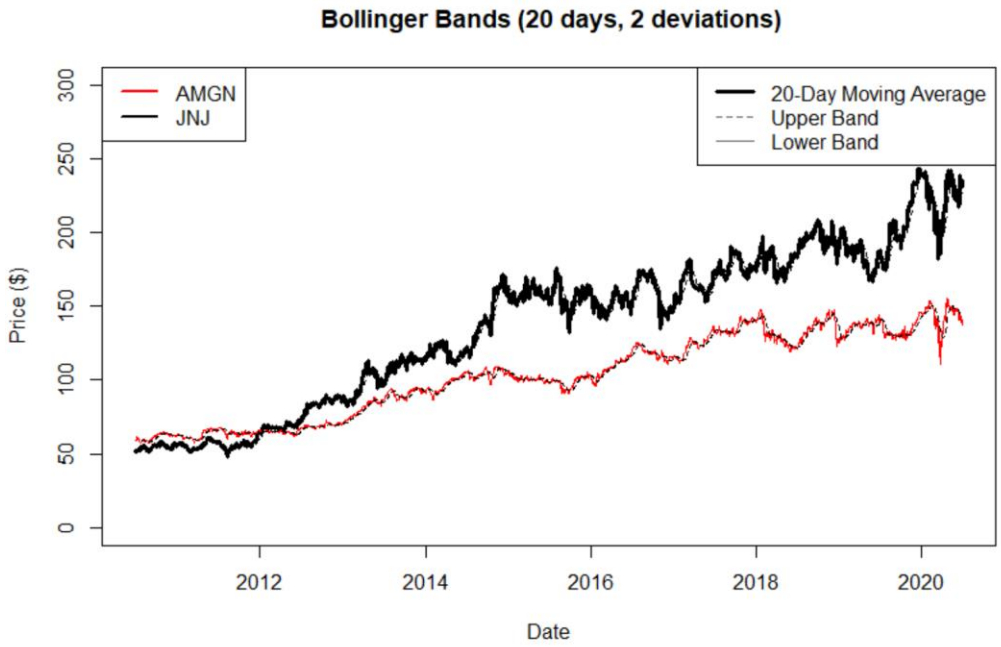
Bollinger Bands Plot

Model Description

We first calculated the 20-day simple moving average (SMA) as demonstrated in the previous step and then we calculated the upper band (two Standard deviations) above the 20-day SMA) and lower band (two std.dev below the 20-day SMA). Then we obtained Closing Price for each stock. The Rollmeanr function was used to calculate the Bollinger Band average and the Rollapply function was used to calculate the Bollinger bands standard deviation. Then we plotted the bands. For example, the BB band for the pharma sector using the technique above is shown below.

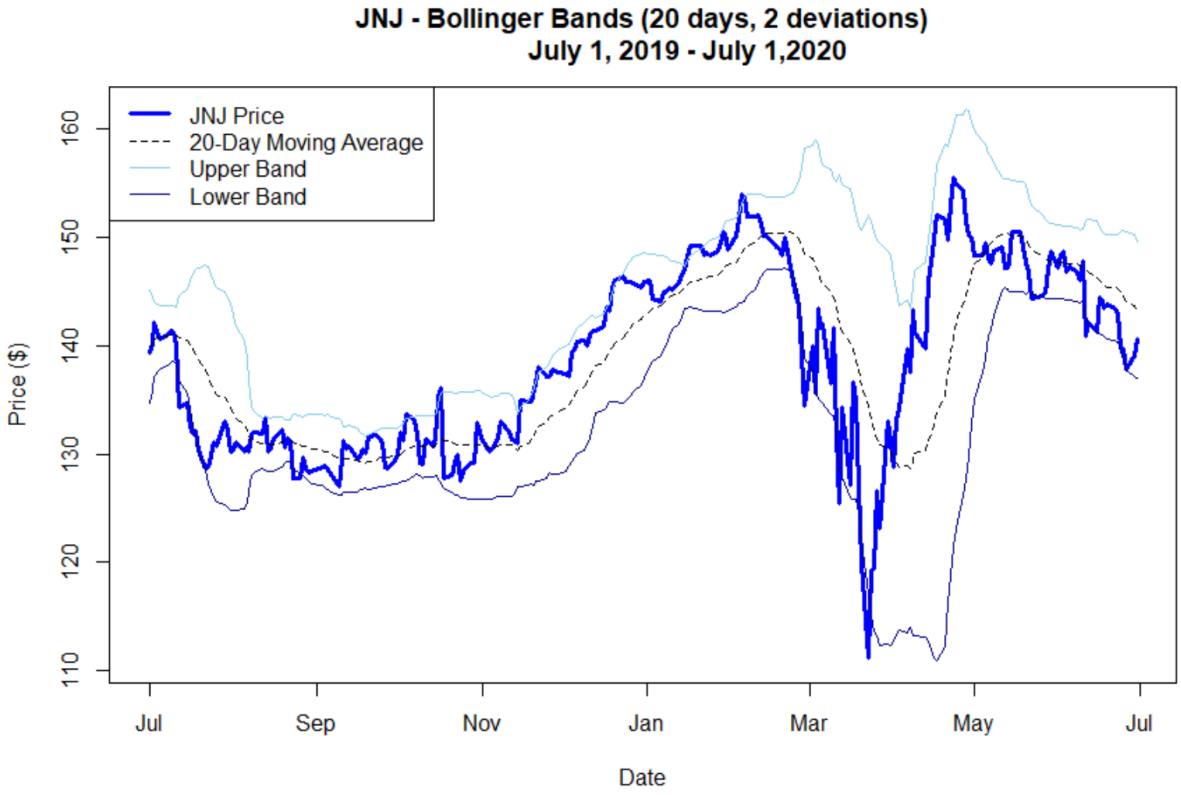
Output Analysis:

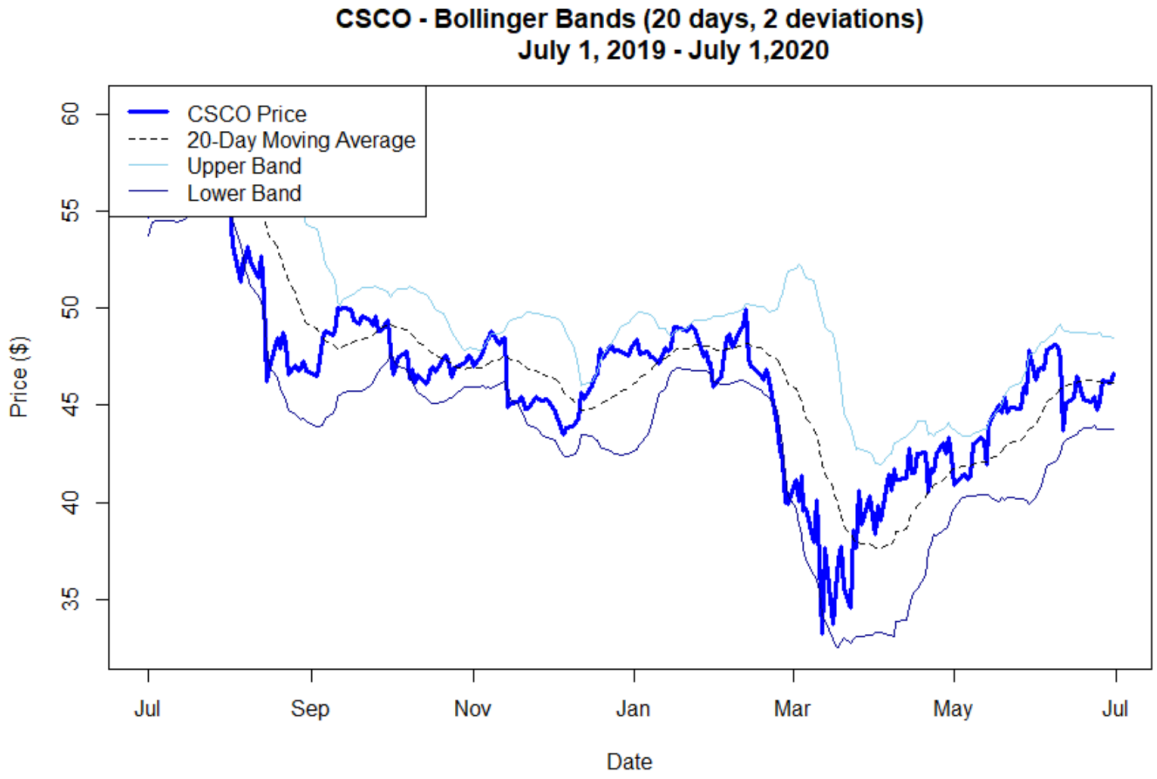
Bollinger Bands

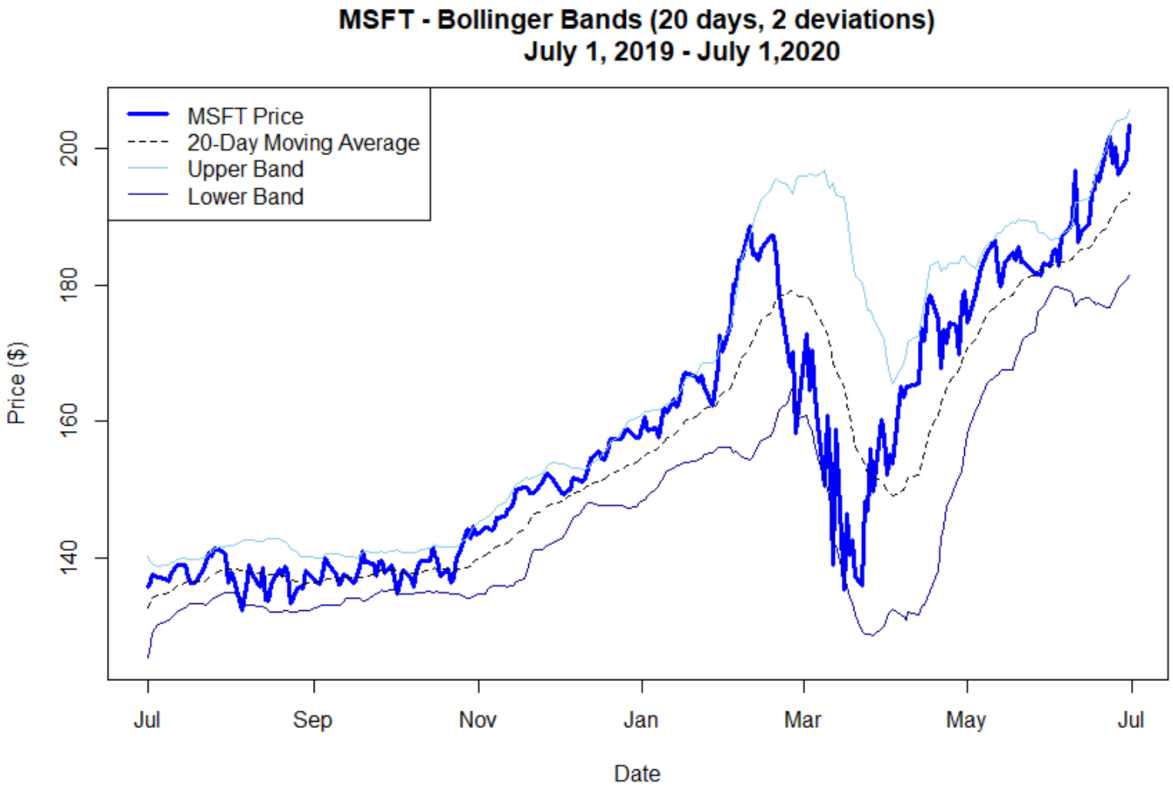


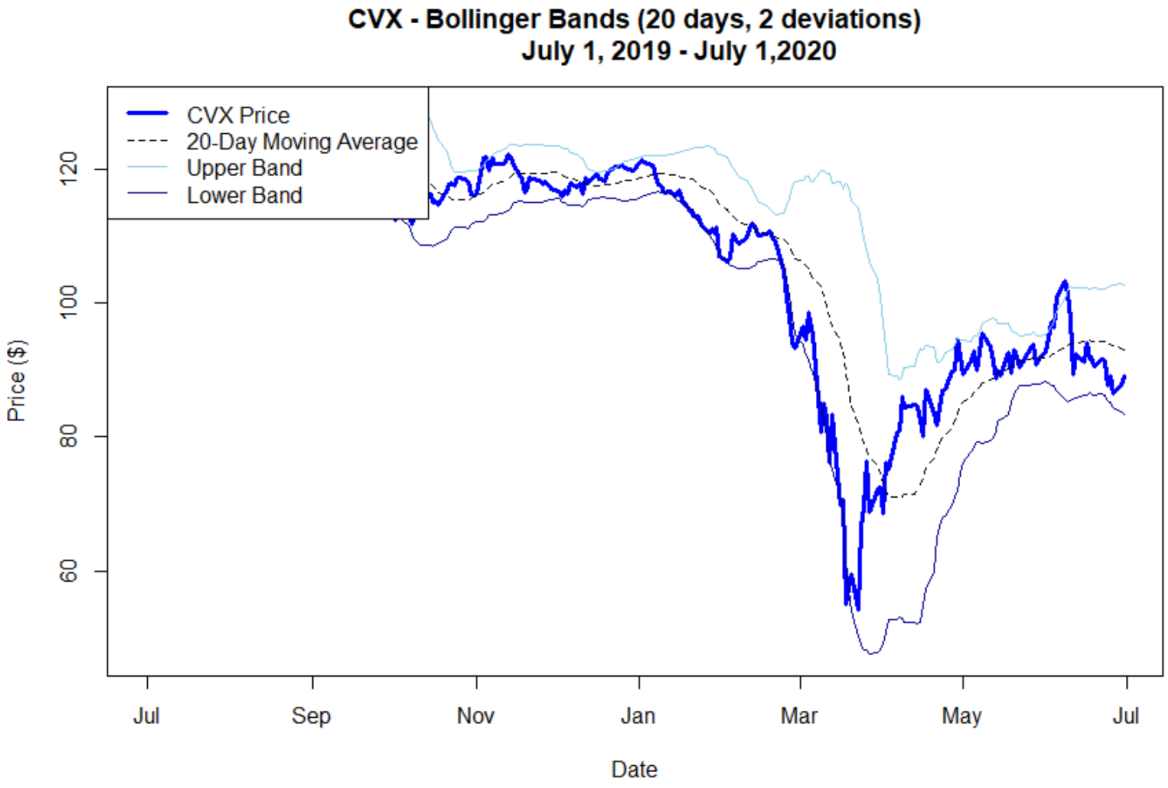
Conclusion/recommendation:

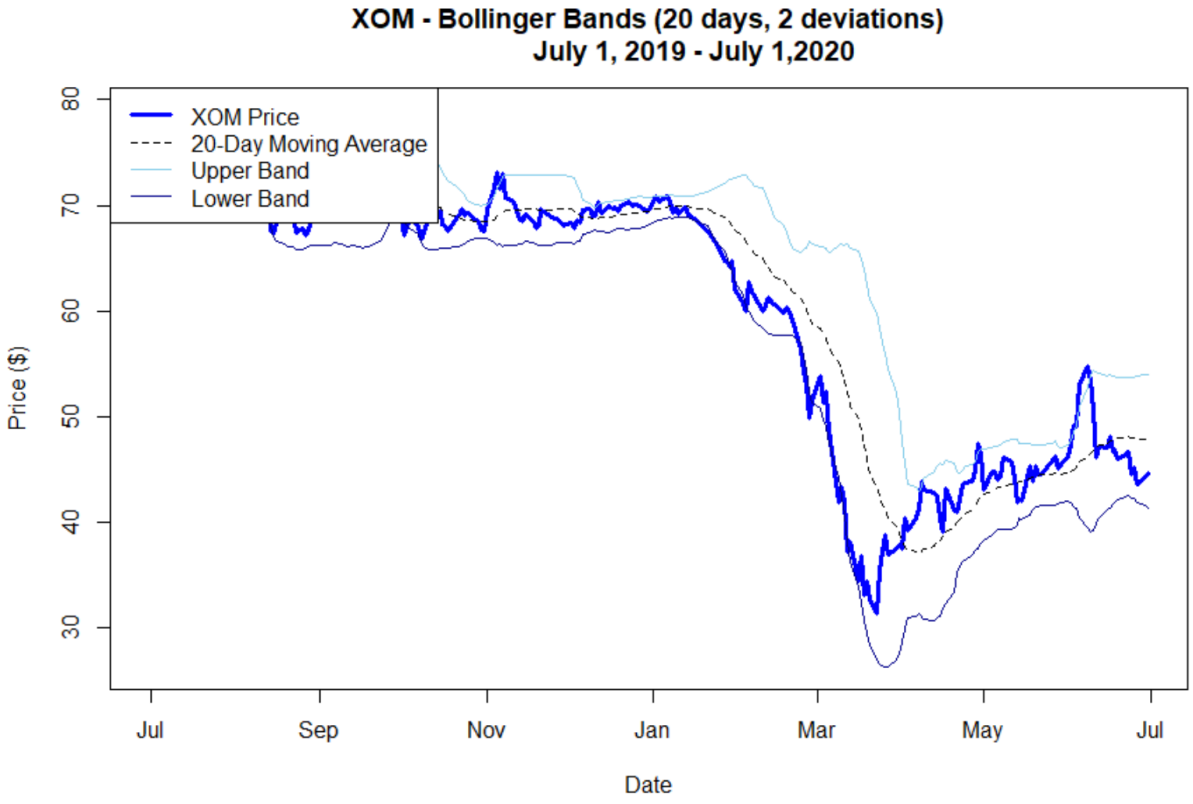
From the above bollinger bands, it can be said that AMGN is less volatile than JNJ.




CSCO - Bollinger Bands (20 days, 2 deviations)
July 1, 2019 - July 1,2020
CSCO Price
20-Day Moving Average
Upper Band
Lower Band
Price ($)
60
55
50
45
40
35
Jul
Sep
Nov
Jan
Mar
May
Jul
Date


MSFT - Bollinger Bands (20 days, 2 deviations)
July 1, 2019 - July 1,2020
MSFT Price
20-Day Moving Average
Upper Band
Lower Band
Price ($)
200
180
160
140
Jul
Sep
Nov
Jan
Mar
May
Jul
Date


CVX - Bollinger Bands (20 days, 2 deviations)
July 1, 2019 - July 1,2020
CVX Price
20-Day Moving Average
Upper Band
Lower Band
Price ($)
120
100
80
60
Jul
Sep
Nov
Jan
Mar
May
Jul
Date


XOM - Bollinger Bands (20 days, 2 deviations)
July 1, 2019 - July 1,2020
XOM Price
20-Day Moving Average
Upper Band
Lower Band
Price ($)
80
70
60
50
40
30
Jul
Sep
Nov
Jan
Mar
May
Jul
Date

Conclusion/recommendation:

From the above Bollinger bands we can see the financial trend of simple moving averages and upper and lower bands and how the prices deflect within them. Tighter Bollinger bands indicate financial decline.

In general, JNJ has much tighter Bollinger bands as compared to AMGN. CVX and XOM have somewhat similar Bollinger bands. Microsoft had a much tighter Bollinger band in the beginning of the 10 year period and a wider band from around 2015 onwards.

Comparing Capital Gains of Multiple Securities Over Time

Model Description:

We calculated total returns for AMGN, JNJ, MSFT and CSCO. Time period = 2019 July 1 to 2020 July 1.

We have merged multiple stocks using the merge function and then were able to combine everything in a single data frame using the cbind function. Then we calculated the return index for each stock by dividing the daily return price with the opening day price. Then the plot was built to visualize the trend. Then we calculate the total returns as well as the logarithmic and cumulative returns. A candlestick chart has been used to describe the same. Gross returns, gross cumulative returns, net cumulative price returns have also been calculated and compared.

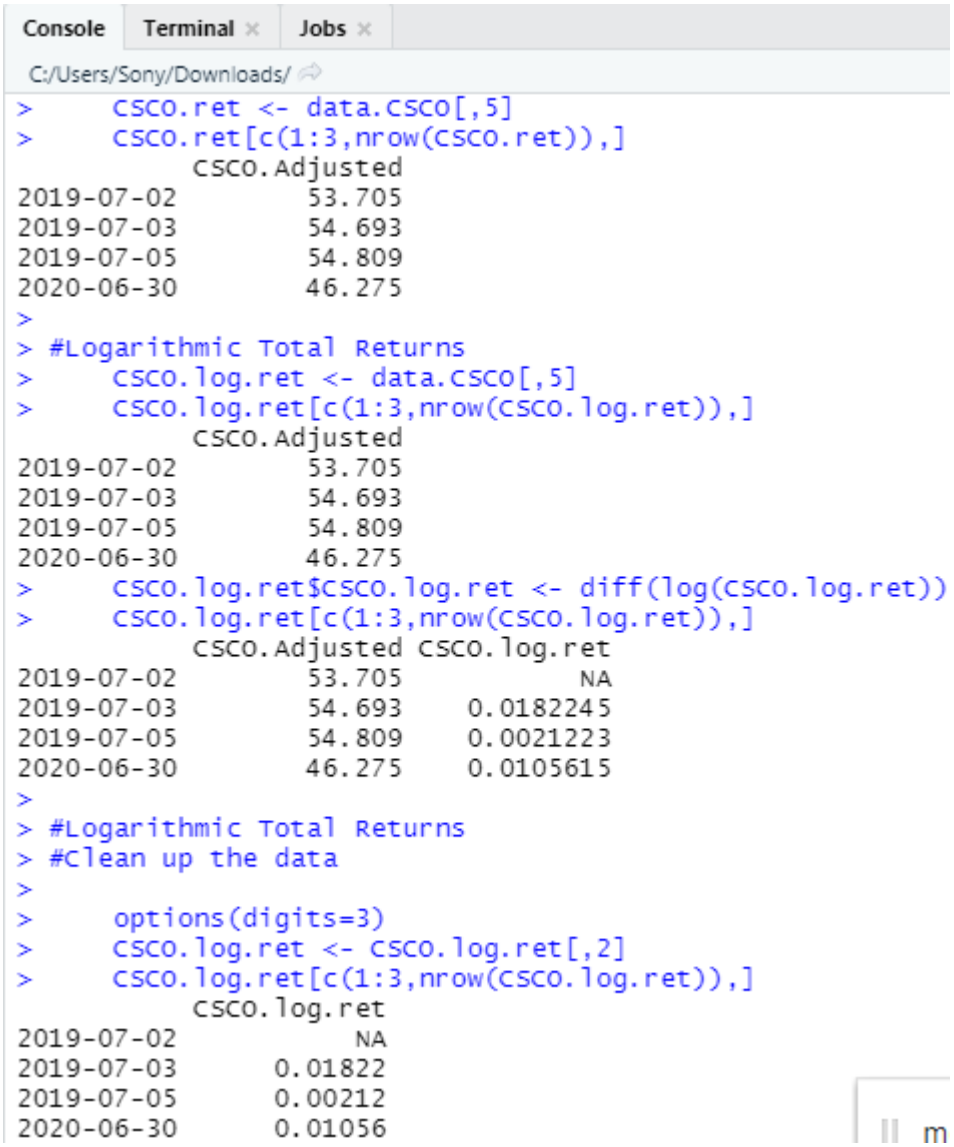
Calculation of total returns:

```
>     #Clean up the data
>     options(digits = 3)
>     AMGN.tot.ret <- AMGN.ret[-1,2]
>     AMGN.tot.ret[c(1:3,nrow(AMGN.tot.ret)),]
           AMGN.tot.ret
2019-07-03       0.0104
2019-07-05      -0.0140
2019-07-08      -0.0199
2020-06-30       0.0186
>
>     options(digits = 3)
>     JNJ.tot.ret <- JNJ.ret[-1,2]
>     JNJ.tot.ret[c(1:3,nrow(JNJ.tot.ret)),]
           JNJ.tot.ret
2019-07-03     0.01507
2019-07-05    -0.01105
2019-07-08     0.00285
2020-06-30     0.01144
>
>     options(digits = 3)
>     CSCO.tot.ret <- CSCO.ret[-1,2]
>     CSCO.tot.ret[c(1:3,nrow(CSCO.tot.ret)),]
           CSCO.tot.ret
2019-07-03      0.01839
2019-07-05      0.00212
2019-07-08     -0.00724
2020-06-30      0.01062
>
>     options(digits = 3)
>     MSFT.tot.ret <- MSFT.ret[-1,2]
>     MSFT.tot.ret[c(1:3,nrow(MSFT.tot.ret)),]
           MSFT.tot.ret
2019-07-03      0.00644
2019-07-05     -0.00291
2019-07-08     -0.00073
2020-06-30      0.02555
>
>     options(digits = 3)
>     CVX.tot.ret <- CVX.ret[-1,2]
>     CVX.tot.ret[c(1:3,nrow(CVX.tot.ret)),]
           CVX.tot.ret
2019-07-03    0.003417
2019-07-05    0.001540
2019-07-08    0.000486
2020-06-30    0.017794
>
>     options(digits = 3)
>     XOM.tot.ret <- XOM.ret[-1,2]
>     XOM.tot.ret[c(1:3,nrow(XOM.tot.ret)),]
           XOM.tot.ret
2019-07-03     0.00951
2019-07-05    -0.00406
2019-07-08     0.00460
2020-06-30     0.00903
```

Console Terminal Jobs

C:/Users/Sony/Downloads/

```
>     JNJ.ret[c(1:3,nrow(JNJ.ret)),]
           JNJ.Adjusted
2019-07-02       136.21
2019-07-03       138.27
2019-07-05       136.74
2020-06-30       140.63
>
> #Logarithmic Total Returns
>
>     JNJ.log.ret <- data.JNJ[,5]
>     JNJ.log.ret[c(1:3,nrow(JNJ.log.ret)),]
           JNJ.Adjusted
2019-07-02       136.21
2019-07-03       138.27
2019-07-05       136.74
2020-06-30       140.63
>     JNJ.log.ret$JNJ.log.ret <- diff(log(JNJ.log.ret))
>     JNJ.log.ret[c(1:3,nrow(JNJ.log.ret)),]
           JNJ.Adjusted JNJ.log.ret
2019-07-02       136.21          NA
2019-07-03       138.27    0.014956
2019-07-05       136.74   -0.011107
2020-06-30       140.63    0.011371
>
>     #Logarithmic Total Returns
>
> #Clean up the data
>     options(digits=3)
>     JNJ.log.ret <- JNJ.log.ret[,2]
>     JNJ.log.ret[c(1:3,nrow(JNJ.log.ret)),]
           JNJ.log.ret
2019-07-02          NA
2019-07-03      0.0150
2019-07-05     -0.0111
2020-06-30      0.0114
>
```



Console | Terminal | Jobs

C:/Users/Sony/Downloads/

```
>     CSCO.ret <- data.CSCO[,5]
>     CSCO.ret[c(1:3,nrow(CSCO.ret)),]
           CSCO.Adjusted
2019-07-02        53.705
2019-07-03        54.693
2019-07-05        54.809
2020-06-30        46.275
>
> #Logarithmic Total Returns
>     CSCO.log.ret <- data.CSCO[,5]
>     CSCO.log.ret[c(1:3,nrow(CSCO.log.ret)),]
           CSCO.Adjusted
2019-07-02        53.705
2019-07-03        54.693
2019-07-05        54.809
2020-06-30        46.275
>     CSCO.log.ret$CSCO.log.ret <- diff(log(CSCO.log.ret))
>     CSCO.log.ret[c(1:3,nrow(CSCO.log.ret)),]
           CSCO.Adjusted CSCO.log.ret
2019-07-02        53.705           NA
2019-07-03        54.693    0.0182245
2019-07-05        54.809    0.0021223
2020-06-30        46.275    0.0105615
>
> #Logarithmic Total Returns
> #Clean up the data
>
>     options(digits=3)
>     CSCO.log.ret <- CSCO.log.ret[,2]
>     CSCO.log.ret[c(1:3,nrow(CSCO.log.ret)),]
           CSCO.log.ret
2019-07-02           NA
2019-07-03      0.01822
2019-07-05      0.00212
2020-06-30      0.01056
```

```
C:/Users/Sony/Downloads/
>     MSFT.ret[c(1:3,nrow(MSFT.ret)),]
           MSFT.Adjusted
2019-07-02        134.92
2019-07-03        135.79
2019-07-05        135.39
2020-06-30        203.51
>
>     #Logarithmic Total Returns
>
>     MSFT.log.ret <- data.MSFT[,5]
>     MSFT.log.ret[c(1:3,nrow(MSFT.log.ret)),]
           MSFT.Adjusted
2019-07-02        134.92
2019-07-03        135.79
2019-07-05        135.39
2020-06-30        203.51
>     MSFT.log.ret$MSFT.log.ret <- diff(log(MSFT.log.ret))
>     MSFT.log.ret[c(1:3,nrow(MSFT.log.ret)),]
           MSFT.Adjusted MSFT.log.ret
2019-07-02        134.92           NA
2019-07-03        135.79    0.0064225
2019-07-05        135.39   -0.0029141
2020-06-30        203.51    0.0252283
>
>     #Logarithmic Total Returns
>
>     #Clean up the data
>     options(digits=3)
>     MSFT.log.ret <- MSFT.log.ret[,2]
>     MSFT.log.ret[c(1:3,nrow(MSFT.log.ret)),]
           MSFT.log.ret
2019-07-02           NA
2019-07-03      0.00642
2019-07-05     -0.00291
2020-06-30      0.02523
>
```

```
C:/Users/Sony/Downloads/
>     XOM.ret <- data.XOM[,5]
>     XOM.ret[c(1:3,nrow(XOM.ret)),]
           XOM.Adjusted
2019-07-02       71.467
2019-07-03       72.147
2019-07-05       71.854
2020-06-30       44.720
>
>     #Logarithmic Total Returns
>
>     XOM.log.ret <- data.XOM[,5]
>     XOM.log.ret[c(1:3,nrow(XOM.log.ret)),]
           XOM.Adjusted
2019-07-02       71.467
2019-07-03       72.147
2019-07-05       71.854
2020-06-30       44.720
>     XOM.log.ret$XOM.log.ret <- diff(log(XOM.log.ret))
>     XOM.log.ret[c(1:3,nrow(XOM.log.ret)),]
           XOM.Adjusted XOM.log.ret
2019-07-02       71.467          NA
2019-07-03       72.147   0.0094638
2019-07-05       71.854  -0.0040638
2020-06-30       44.720   0.0089848
>
>     #Logarithmic Total Returns
>
>     #Clean up the data
>     options(digits=3)
>     XOM.log.ret <- XOM.log.ret[,2]
>     XOM.log.ret[c(1:3,nrow(XOM.log.ret)),]
           XOM.log.ret
2019-07-02          NA
2019-07-03     0.00946
2019-07-05    -0.00406
2020-06-30     0.00898
>
```

Conclusion/recommendation:

From the above result, it is clear that Amgen total returns have increased with time. JNJ total returns have fluctuated and decreased overall with time. Cisco total returns also fluctuated and eventually decreased with time. Microsoft total returns have consistently increased with time. CVX total returns have overall increased with time. XOM total returns first decreased and then increased with time.

Logarithmic returns will be used for cumulative returns

Amgen's logarithmic returns:

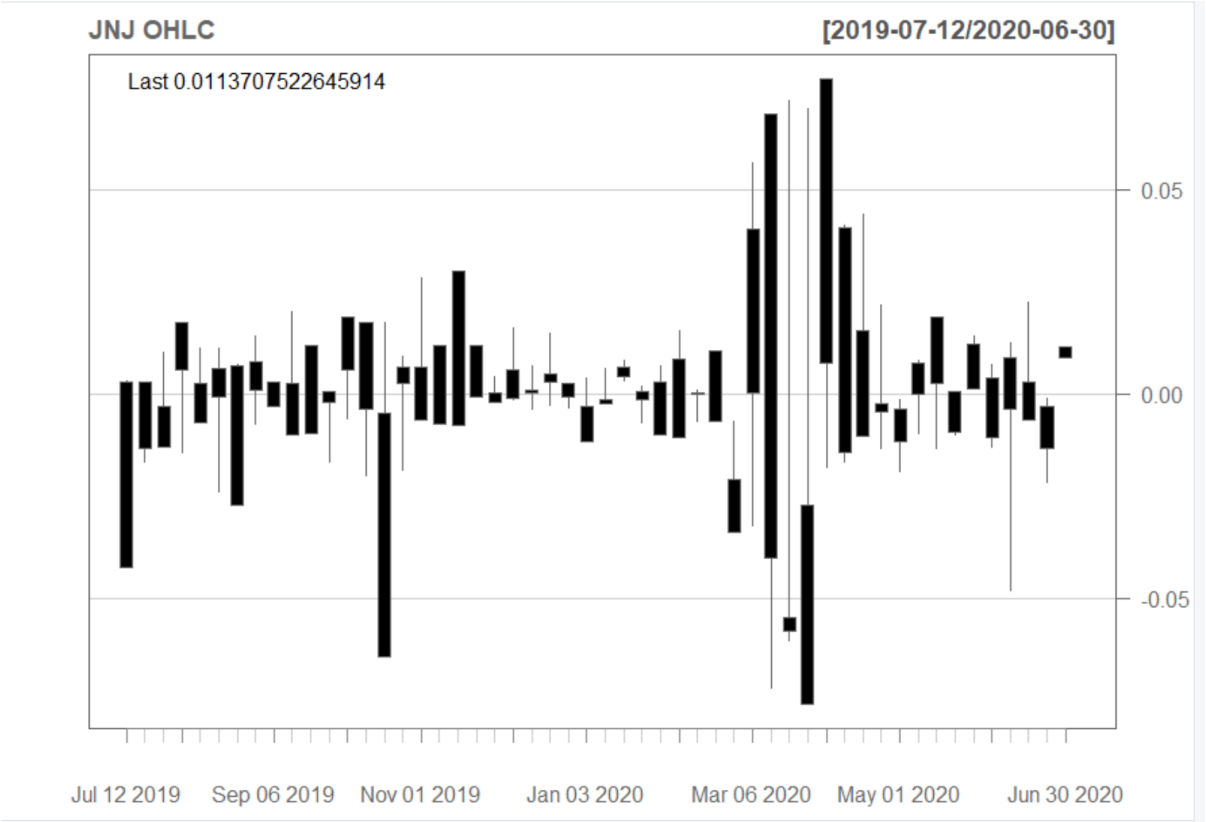
```
> AMGN.log.ret[c(1:3,nrow(AMGN.log.ret)),]
           AMGN.Adjusted AMGN.log.ret
2019-07-02        181.58           NA
2019-07-03        183.46     0.010341
2019-07-05        180.90    -0.014098
2020-06-30        235.86     0.018399
```

Amgen Candlestick chart for returns: 2019 - 2020 by month.

Conclusion/recommendation:

From the above logarithmic returns, it is evident that from the stock returns have increased from 07-02-2019 until 07-03-2019 and then decreased from 07-03-2019 to 07-05-2019. Further the stock value increased from 07-05-2019 to 06-30-2020.

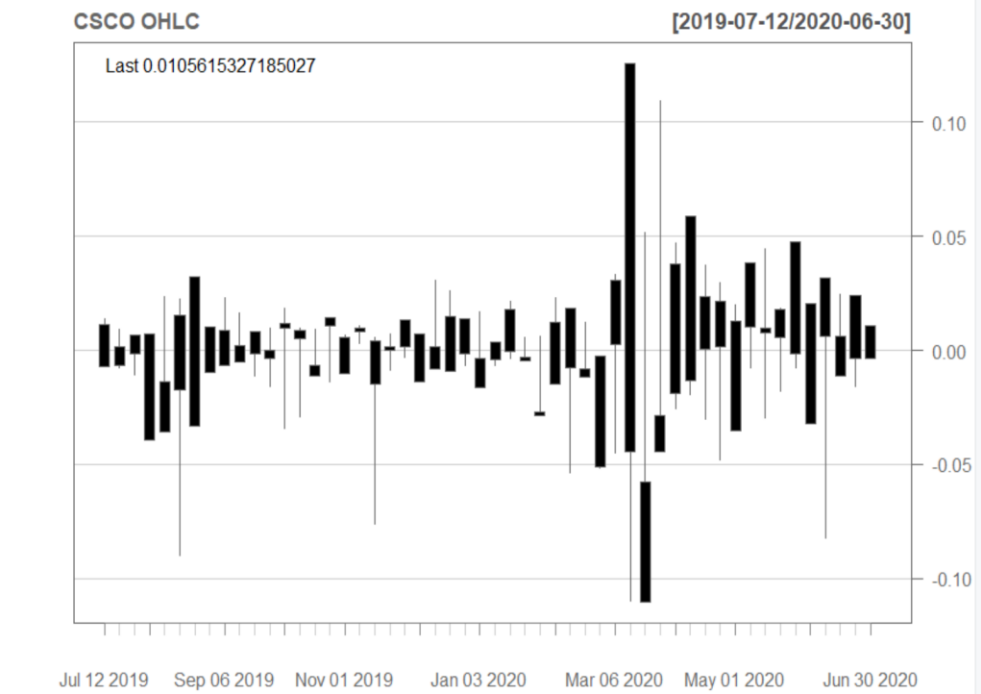
Johnson & Johnson - Candlestick chart:



Conclusion/recommendation:

From the above candlestick chart of JNJ, we can see that around Nov 2019 and April 2020 shows a significantly lower opening price while around March and beginning of May 2020, we can see a higher closing price overall.

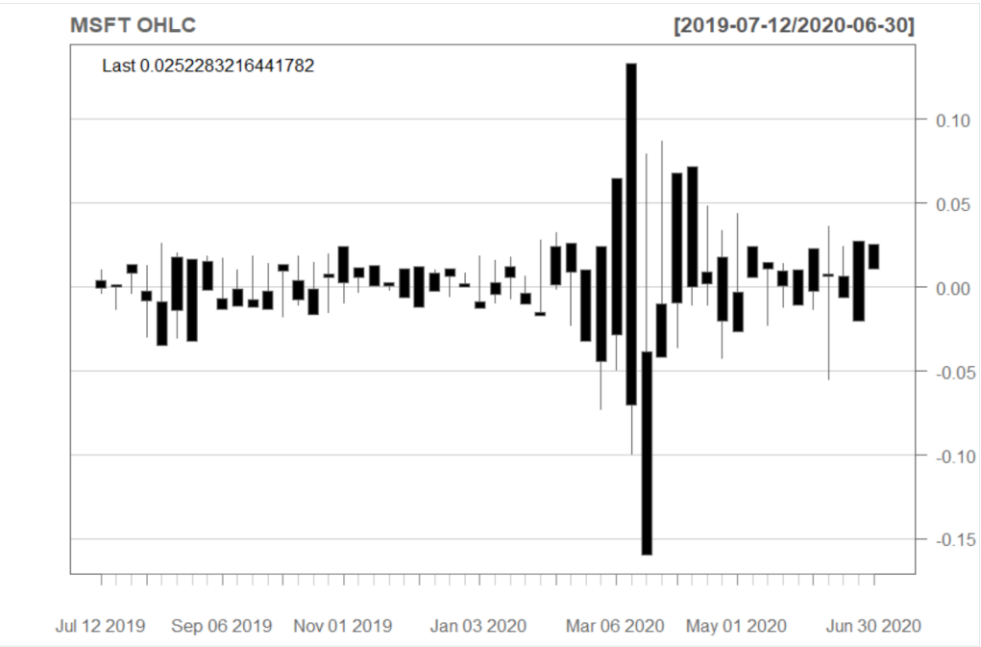
CSCO Candlestick chart:



Conclusion/recommendation:

From the above candlestick chart, we can see Cisco saw a much higher opening price than average during March 2020 and then a much lower opening and closing price than usual in April 2020.

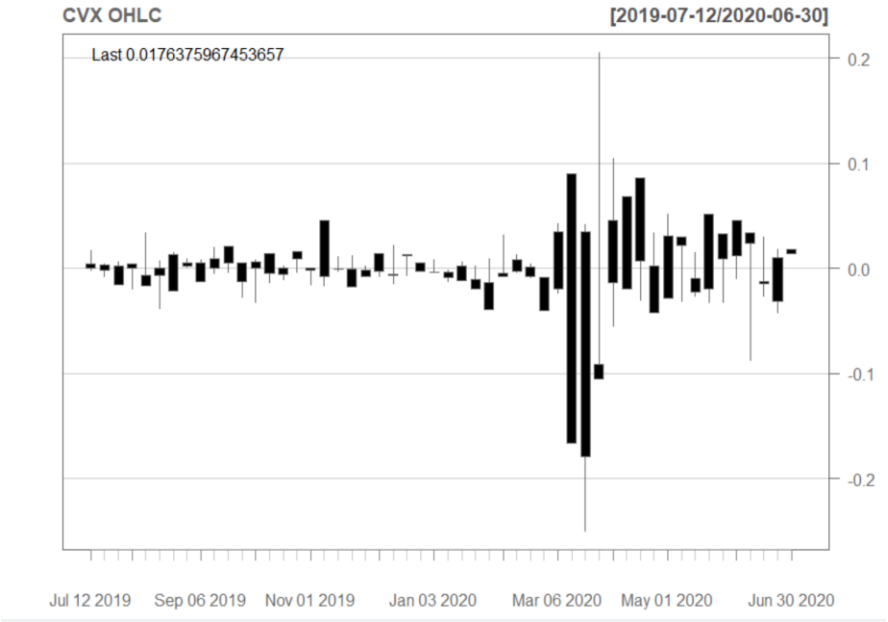
MSFT:



Conclusion/recommendation:

From the above candlestick chart, we can infer that MSFT saw a much higher opening price in March followed by a much lower opening as well as closing price around the end of March 2020.

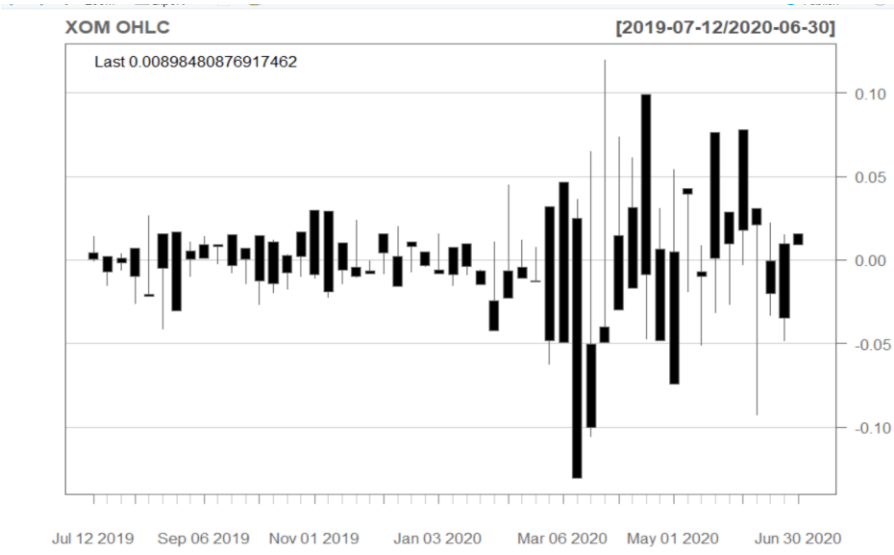
CVX:



Conclusion/recommendation:

From the above chart, it is evident that overall CVX saw comparatively lower closing prices around March 2020.

XOM:



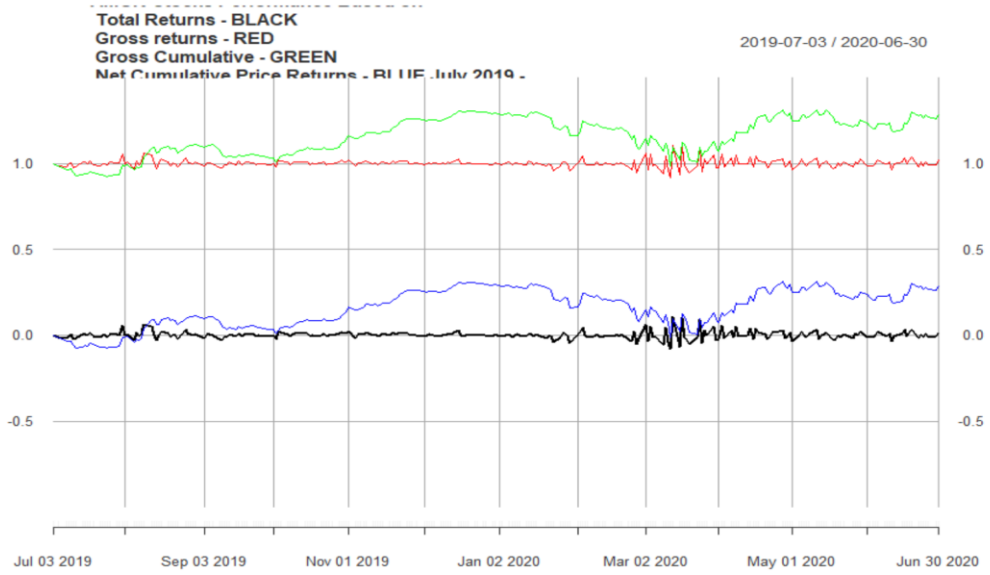
Conclusion/recommendation:

From the above candlestick chart, it is clear that XOM saw a relatively higher opening price around May 2020 and a comparatively lower closing price around March 2020.

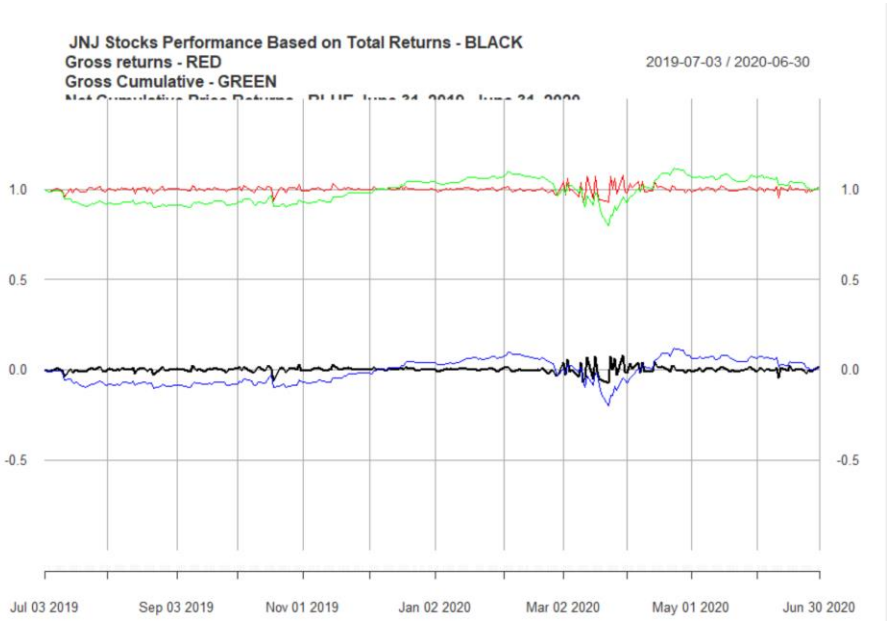
Comparison of the 6 stocks in the portfolio (AMGN, JNJ, MSFT, CSCO, CVX, XOM):

Output Analysis:

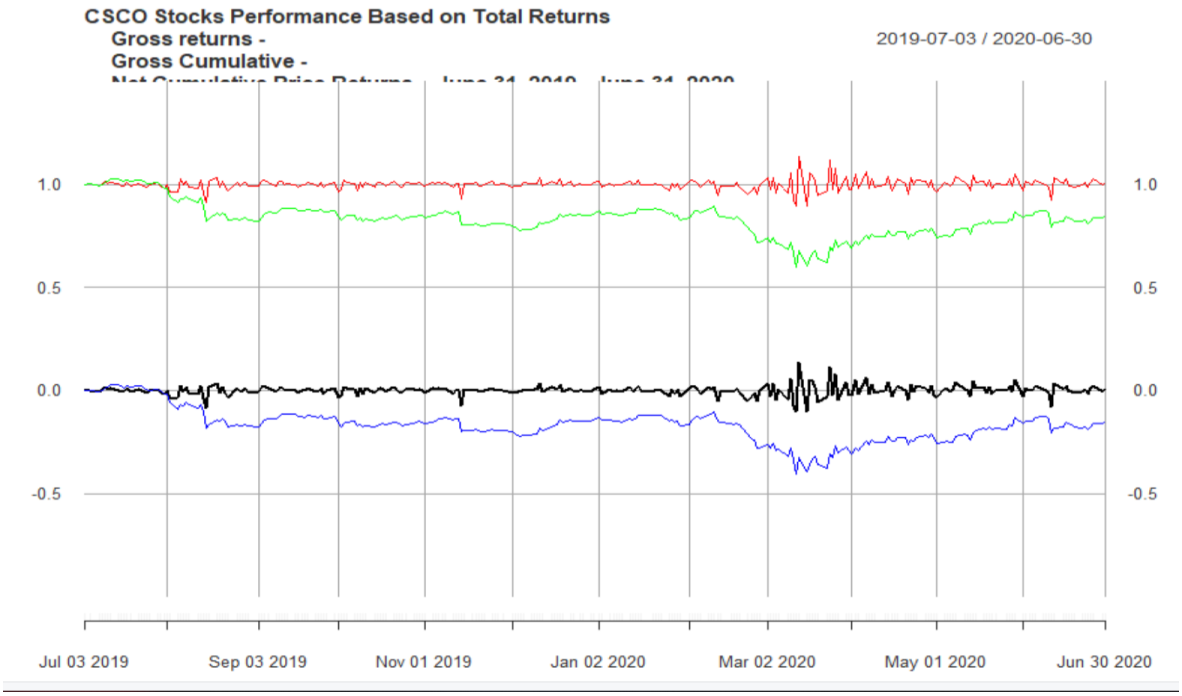
AMGN:



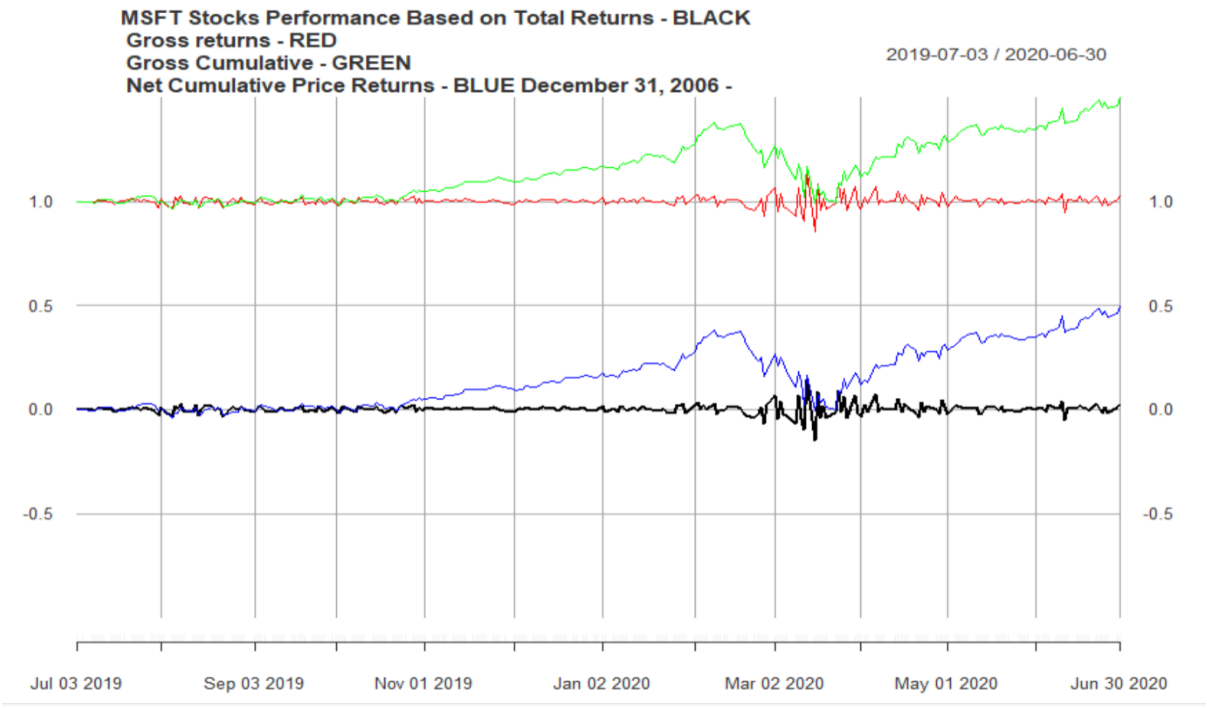
JNJ:

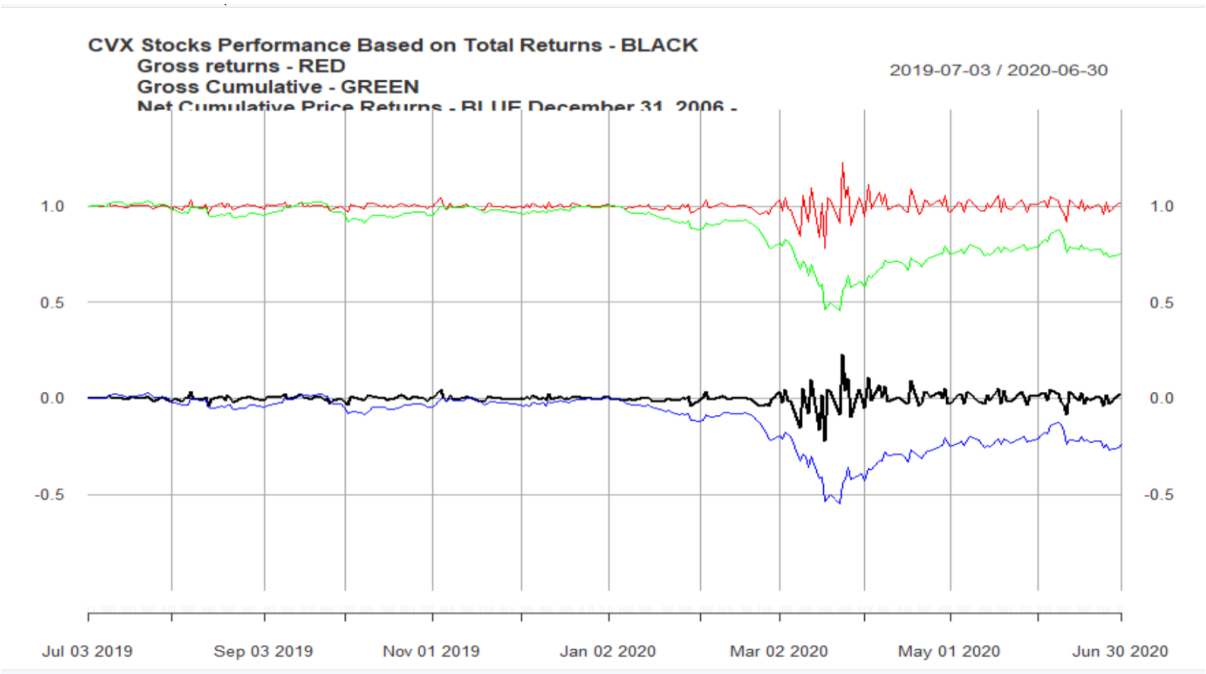


CSCO:

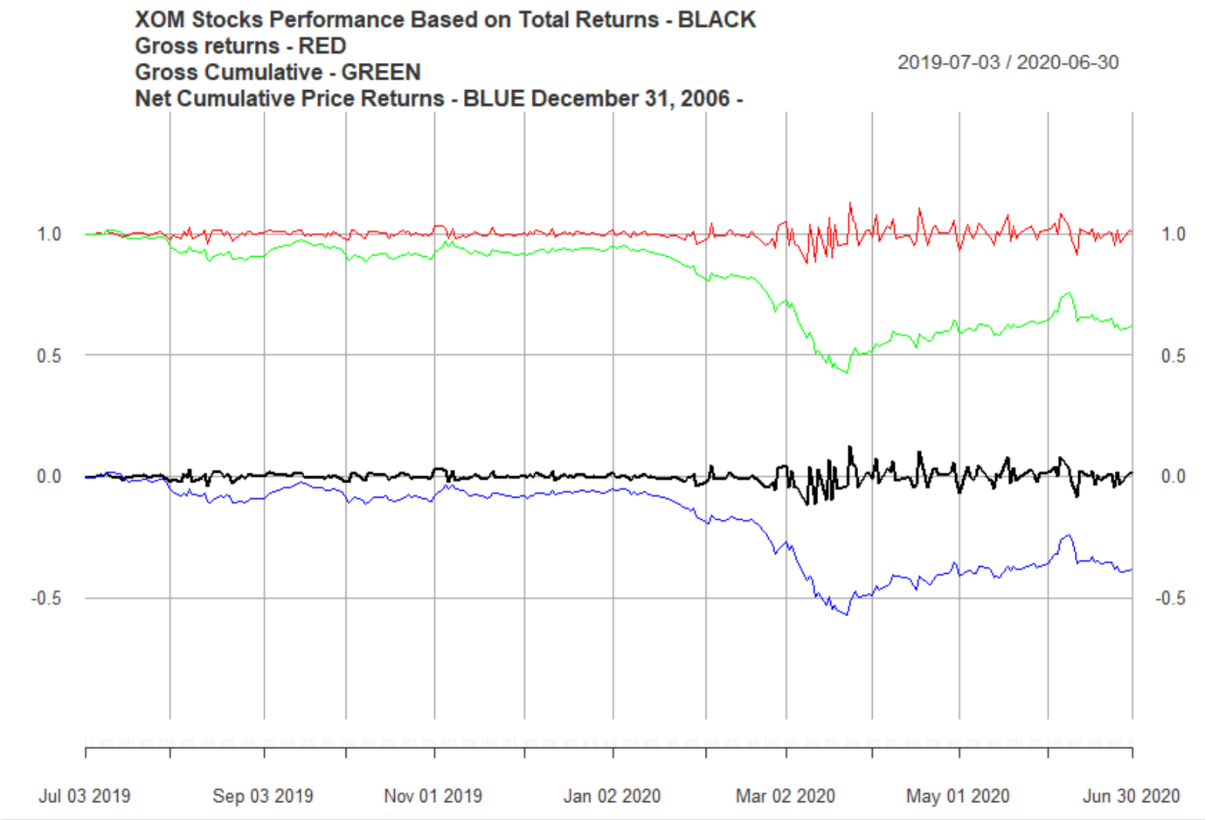

CSCO Stocks Performance Based on Total Returns
Gross returns -
Gross Cumulative -
2019-07-03 / 2020-06-30
1.0
0.5
0.0
-0.5
Jul 03 2019
Sep 03 2019
Nov 01 2019
Jan 02 2020
Mar 02 2020
May 01 2020
Jun 30 2020

MSFT:


MSFT Stocks Performance Based on Total Returns - BLACK
Gross returns - RED
Gross Cumulative - GREEN
Net Cumulative Price Returns - BLUE December 31, 2006 -
2019-07-03 / 2020-06-30
1.0
0.5
0.0
-0.5
Jul 03 2019
Sep 03 2019
Nov 01 2019
Jan 02 2020
Mar 02 2020
May 01 2020
Jun 30 2020



XOM:



Conclusion/recommendation:

We have plotted the total returns, gross returns, gross cumulative and net cumulative price returns against time, for all the 6 stocks in the graphs above. Gross

cumulative and net cumulative plots show a similar pattern for all the 6 stocks and so do total returns and gross returns. Amgen stocks show a dip for all variables around March 2020, however, the variables for JNJ remain fairly consistent for the entire duration of July 2019 to June 2020. In case of Cisco, we see a slight dip around March 2020 but in case of Microsoft, we see that the stocks show a dramatic dip around March 2020 but have rebounded after April 2020 to all time high. CVX and XOM have a pretty similar graph showing a dip around March 2020 and the stocks have remained low since then.

Cumulative returns Comparison:

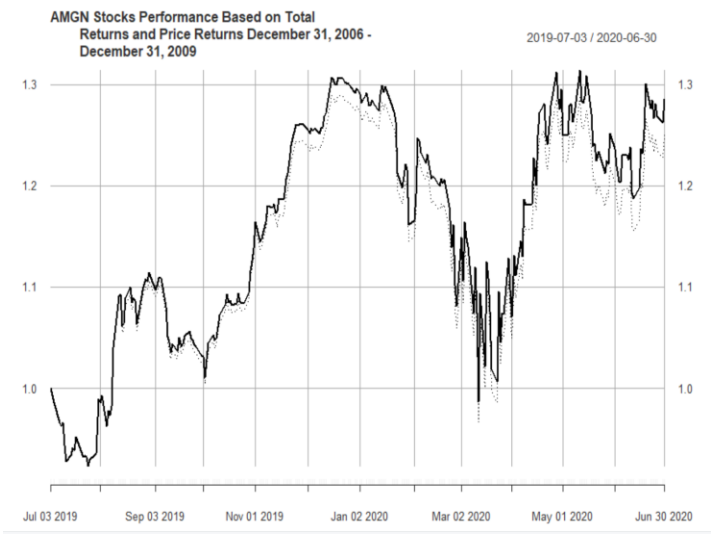
```
> AMGN.cumret=exp(logcumret)-1
> AMGN.cumret
[1] 0.508378
> JNJ.cumret=exp(logcumret)-1
> JNJ.cumret
[1] 0.508378
> CSCO.cumret=exp(logcumret)-1
> CSCO.cumret
[1] 0.508378
> MSFT.cumret=exp(logcumret)-1
> MSFT.cumret
[1] 0.508378
>
```

Fluctuation of stock return prices:

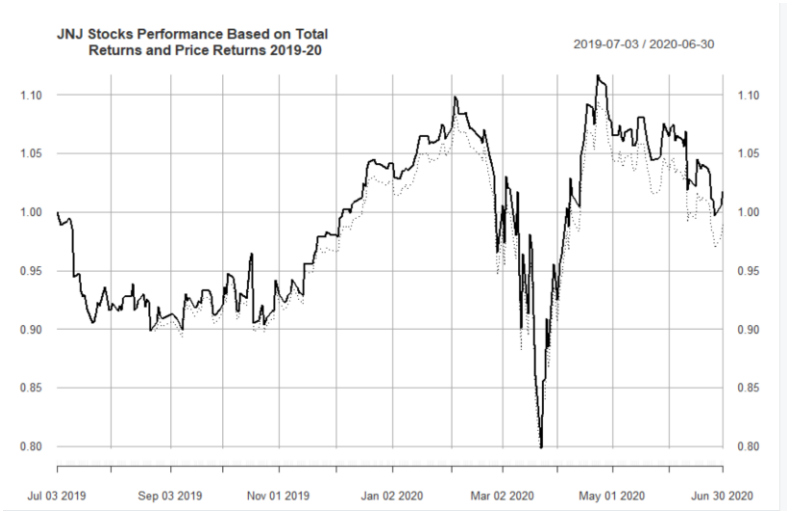
```
> MSFT.y.range
[1] 0.9618071 1.4987216
> AMGN.y.range
[1] 0.9232686 1.3141950
> JNJ.y.range
[1] 0.7819052 1.1169744
> MSFT.y.range
[1] 0.9618071 1.4987216
> CSCO.y.range
[1] 0.5878187 1.0277975
```

Stock performance analysis:

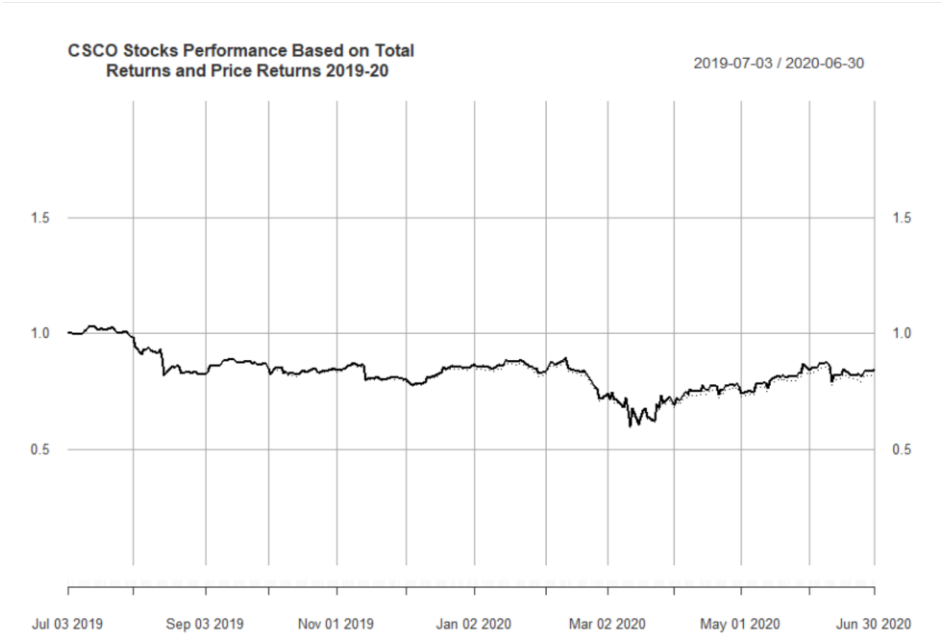
Amgen:


AMGN Stocks Performance Based on Total Returns and Price Returns December 31, 2006 - December 31, 2009
2019-07-03 / 2020-06-30
1.3
1.2
1.1
1.0
Jul 03 2019
Sep 03 2019
Nov 01 2019
Jan 02 2020
Mar 02 2020
May 01 2020
Jun 30 2020

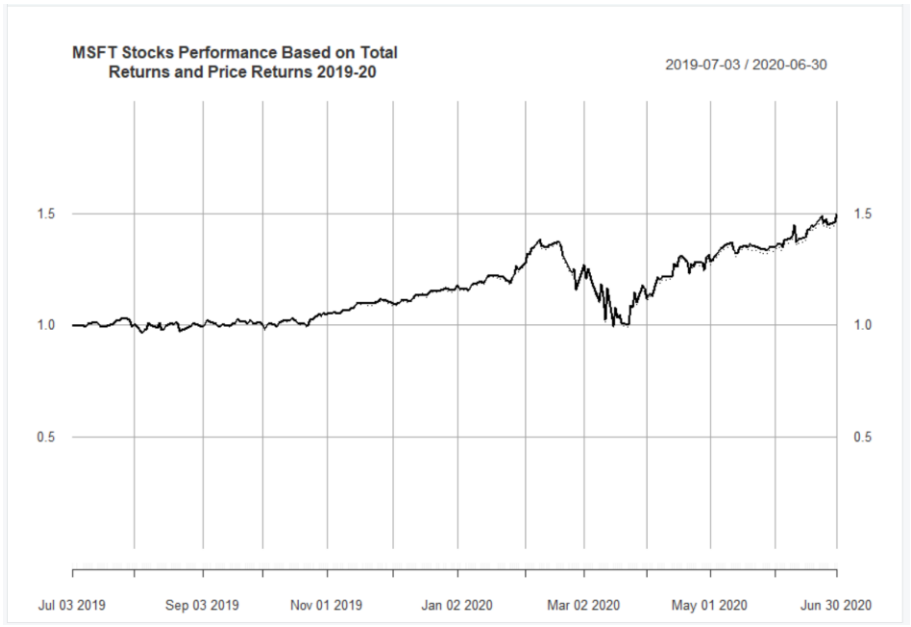
JNJ


JNJ Stocks Performance Based on Total Returns and Price Returns 2019-20
2019-07-03 / 2020-06-30
1.10
1.05
1.00
0.95
0.90
0.85
0.80
Jul 03 2019
Sep 03 2019
Nov 01 2019
Jan 02 2020
Mar 02 2020
May 01 2020
Jun 30 2020

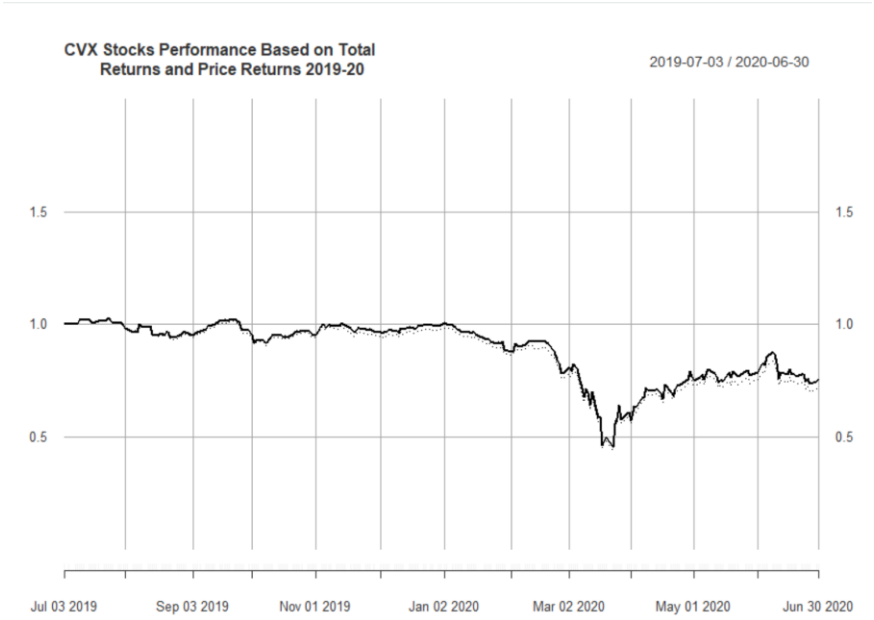
CSCO


CSCO Stocks Performance Based on Total Returns and Price Returns 2019-20
2019-07-03 / 2020-06-30
1.5
1.0
0.5
Jul 03 2019
Sep 03 2019
Nov 01 2019
Jan 02 2020
Mar 02 2020
May 01 2020
Jun 30 2020

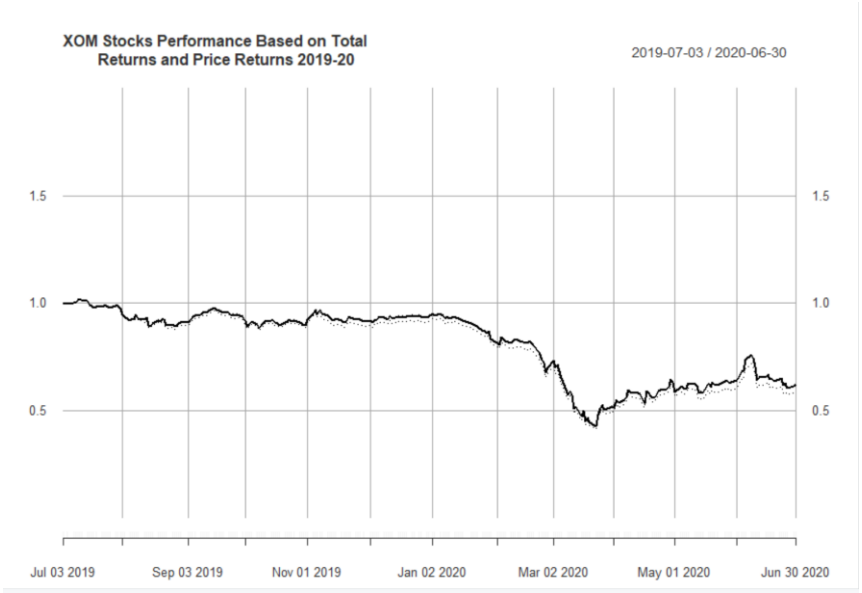
MSFT:


MSFT Stocks Performance Based on Total Returns and Price Returns 2019-20
2019-07-03 / 2020-06-30
1.5
1.0
0.5
Jul 03 2019
Sep 03 2019
Nov 01 2019
Jan 02 2020
Mar 02 2020
May 01 2020
Jun 30 2020

CVX:



XOM:



Conclusion/recommendation:

We have plotted the stock performance based on total returns and price returns for all our 6 stocks for the time period ranging from July 2019 to June 2020.

In case of Amgen, we can see from the graph that the stocks have consistently risen and fallen and that there is a lot of variation - they were at an all time high during January, May and June 2020. There was a steep drop around March 2020 and this can be attributed to the market situation created by COVID-19 pandemic.

In the case of JNJ, the stocks shot up in February 2020, followed by a steep drop in March 2020 and then followed by an equally steep rebound in May 2020. This dramatic fluctuation can also be attributed to COVID-19 pandemic as this is a pharma company.

In case of Cisco, the stocks seemed to have a consistent performance with a slight drop around March 2020. Microsoft's stocks started doing a lot better around Jan 2020 but had to suffer a drop around March 2020 but bounced back up to normal after April 2020.

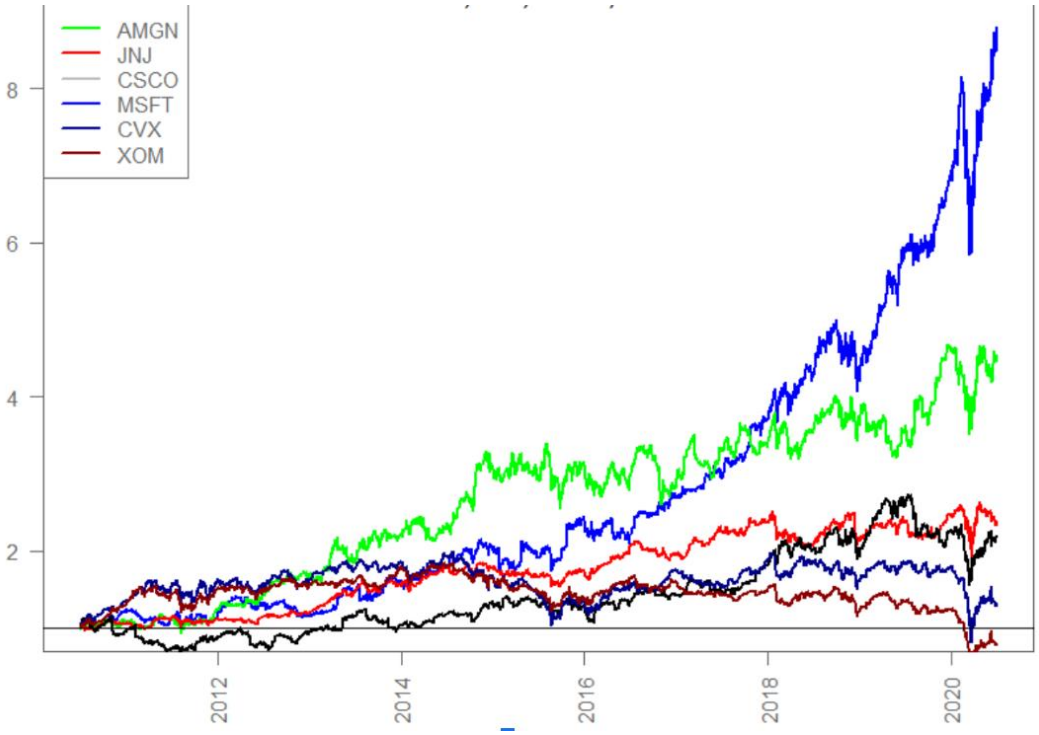
Both CVX and XOM have very similar graphs and they both seem to have suffered a drop around March 2020, due to COVID-19 travel restrictions and have not been able to bounce back up since then.

Portfolio Return:

Model Description:

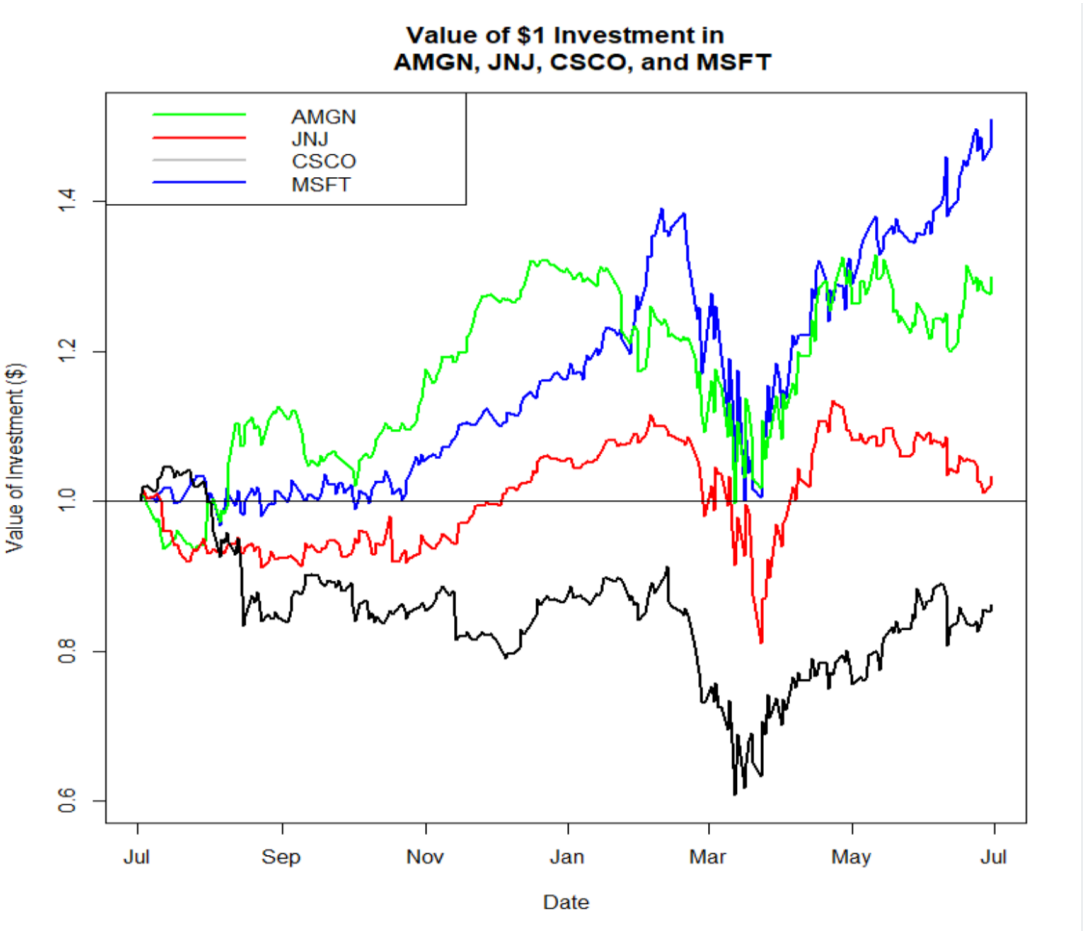
The portfolio return graph shows the value of a $1 investment in each stock for the time period chosen for this study. This clearly describes the trend of investment in each sector. Individual stocks will be normalized and combined so that we can assess investment values by sector. We used the merge function to bind all securities under one data frame and then calculated the index values by dividing the daily index with the index value as of July 1st 2010.

Maximum/Minimum of each security:



Findings:

MSFT has the best portfolio return among all 6 stocks and sectors. AMGN came in second. Energy sector performed the worst.



Model Description:

We calculated the First and Last Adjusted Closing Price for each security over the investment period AMGN, CSCO, MSFT, JNJ. Then we calculated returns for each security over the investment period. Since there are only two observations in "rets", the Delt command was used. "lapply" was used to apply "delt" to all four securities. We calculated the weight of each security in the portfolio and then we created variables representing the dollar amount of investment in each security. Next, we had to calculate the weight by dividing the amount invested by the total investment ($100k). Here the split in investment is AMGN 50K + MSFT 10K + APPL 30K + JNJ 10K (total 100K). Lastly, we calculated portfolio return (weights x return).

Constructing Benchmark Portfolio Returns

Model Description:

The benchmark is used to compare against a portfolio to measure performance. We calculated total returns of each security using the adjusted close. Then we converted to data. Frame object and subset data. In an Equal-Weighted Portfolio, the indexes are balanced quarterly which means that in

between quarters the constituent weights can fluctuate based on their performance. On each rebalancing date, the weights of each firm were set to reset to 1/N (number of securities in the portfolio). Next, we calculated the index value for each security during the quarter and combined quarterly EW portfolio values into one data object. In our analysis, we have taken the weights to be 16667 for each of our 6 stocks.

Value - Weighted portfolio

Model Description:

In a VW portfolio the returns of larger firms are given more weight. In this project, weights are rebalanced at the start of each quarter using the prior quarter end's capitalization data. The calculation of market capitalization of each security has been included in the portfolio. Market cap is equal to the price multiplied by the outstanding shares. The date and PRICE.qtr are merged using a combination of the na.locf and merge commands. The merge command combines two data objects. Finally, pie charts of the weight are created to better describe the distribution of the VW portfolio. Normalized EW and VW portfolio price charts are generated to compare the capital gains by using the two methods.

Calculation of portfolio returns

```
> # Calculate portfolio return (weights x return)
>   port.ret.4asset <-
+   w.AMGN*rets$AMGN+ w.JNJ*rets$JNJ + w.CSCO*rets$CSCO + w.MSFT*rets$MSFT+w.CVX*rets$CVX+w.XOM*rets$XO
M
>   port.ret.4asset
[1] 331
>
```

Return matrix

```
>    ret <- c(rets$AMGN,rets$JNJ,rets$CSCO,rets$MSFT, rets$CVX, rets$XOM)
>    mat.ret <- matrix(ret,6)
>    mat.ret
         [,1]
[1,]  461.1
[2,]  220.7
[3,]  186.1
[4,] 1012.3
[5,]   93.4
[6,]   11.4
```

Benchmark portfolio for 2019-2020

```
> # Create object with only the relevant data
>     port <- data.AMGN[,c(4,5)]
>     port <- merge(port,data.JNJ[,c(4,5)])
>
>     port <- merge(port,data.CSCO[,c(4,5)])
>     port <- merge(port,data.MSFT[,c(4,5)])
>
>     port <- merge(port,data.CVX[,c(4,5)])
>     port <- merge(port,data.XOM[,c(4,5)])
>
>
>     port[c(1:3,nrow(port)),]
           AMGN.Close AMGN.Adjusted JNJ.Close JNJ.Adjusted CSCO.Close CSCO.Adjusted MSFT.Close
2019-07-02        187           182       140          136       55.8          53.7        137
2019-07-03        189           183       142          138       56.5          54.7        137
2019-07-05        186           181       141          137       56.6          54.8        137
2020-06-30        236           236       141          141       46.6          46.3        204
           MSFT.Adjusted CVX.Close CVX.Adjusted XOM.Close XOM.Adjusted
2019-07-02           135     122.9        117.4      75.7         71.5
2019-07-03           136     123.3        117.8      76.4         72.1
2019-07-05           135     123.5        118.0      76.1         71.9
2020-06-30           204      89.2         89.2      44.7         44.7
>
```

Step 2: Calculate total returns of each security using Adjusted.Close

```
>     port[c(1:3,nrow(port)),]
           AMGN.Close AMGN.Adjusted JNJ.Close JNJ.Adjusted CSCO.Close CSCO.Adjusted MSFT.Close
2019-07-02        187           182       140          136       55.8          53.7        137
2019-07-03        189           183       142          138       56.5          54.7        137
2019-07-05        186           181       141          137       56.6          54.8        137
2020-06-30        236           236       141          141       46.6          46.3        204
           MSFT.Adjusted CVX.Close CVX.Adjusted XOM.Close XOM.Adjusted AMGN.ret JNJ.ret CSCO.ret
2019-07-02           135     122.9        117.4      75.7         71.5       NA      NA       NA
2019-07-03           136     123.3        117.8      76.4         72.1   0.0104  0.0151  0.01839
2019-07-05           135     123.5        118.0      76.1         71.9  -0.0140 -0.0110  0.00212
2020-06-30           204      89.2         89.2      44.7         44.7   0.0186  0.0114  0.01062
           MSFT.ret CVX.ret  XOM.ret
2019-07-02       NA      NA       NA
2019-07-03  0.00644 0.00342  0.00951
2019-07-05 -0.00291 0.00154 -0.00406
2020-06-30  0.02555 0.01779  0.00903
```

Step 3: Convert to data frame and subset the required data

```
> # Convert to data.frame object and subset data from Dec 31, 2006 (to include data from 3rd yr of port
folio)
> # to Dec 31, 2019 leaving the Dec 31, 2009 data in because it will
> # be used for later calculations.
>     port <- cbind(data.frame(index(port)),
+                   data.frame(port))
>     names(port)[1] <- paste("date")
>     port[c(1:3,nrow(port)),]
                 date AMGN.Close AMGN.Adjusted JNJ.Close JNJ.Adjusted CSCO.Close CSCO.Adjusted
2019-07-02 2019-07-02        187           182       140          136       55.8          53.7
2019-07-03 2019-07-03        189           183       142          138       56.5          54.7
2019-07-05 2019-07-05        186           181       141          137       56.6          54.8
2020-06-30 2020-06-30        236           236       141          141       46.6          46.3
           MSFT.Close MSFT.Adjusted CVX.Close CVX.Adjusted XOM.Close XOM.Adjusted AMGN.ret JNJ.ret
2019-07-02        137           135     122.9        117.4      75.7         71.5       NA      NA
2019-07-03        137           136     123.3        117.8      76.4         72.1   0.0104  0.0151
2019-07-05        137           135     123.5        118.0      76.1         71.9  -0.0140 -0.0110
2020-06-30        204           204      89.2         89.2      44.7         44.7   0.0186  0.0114
           CSCO.ret MSFT.ret CVX.ret  XOM.ret
2019-07-02       NA       NA      NA       NA
2019-07-03  0.01839  0.00644 0.00342  0.00951
2019-07-05  0.00212 -0.00291 0.00154 -0.00406
2020-06-30  0.01062  0.02555 0.01779  0.00903
>
```

Step 4: Select the required data

```
>     #Selecting data from 2019 and 2020
>
>     port <- subset(port,
+                    port$date >= "2019-07-03")
>     port[c(1:3,nrow(port)),]
                date AMGN.Close AMGN.Adjusted JNJ.Close JNJ.Adjusted CSCO.Close CSCO.Adjusted
2019-07-03 2019-07-03        189           183       142          138       56.5          54.7
2019-07-05 2019-07-05        186           181       141          137       56.6          54.8
2019-07-08 2019-07-08        182           177       141          137       56.2          54.4
2020-06-30 2020-06-30        236           236       141          141       46.6          46.3
           MSFT.Close MSFT.Adjusted CVX.Close CVX.Adjusted XOM.Close XOM.Adjusted AMGN.ret  JNJ.ret
2019-07-03        137           136     123.3        117.8      76.4         72.1   0.0104  0.01507
2019-07-05        137           135     123.5        118.0      76.1         71.9  -0.0140 -0.01105
2019-07-08        137           135     123.6        118.0      76.5         72.2  -0.0199  0.00285
2020-06-30        204           204      89.2         89.2      44.7         44.7   0.0186  0.01144
           CSCO.ret MSFT.ret  CVX.ret  XOM.ret
2019-07-03  0.01839  0.00644 0.003417  0.00951
2019-07-05  0.00212 -0.00291 0.001540 -0.00406
2019-07-08 -0.00724 -0.00073 0.000486  0.00460
2020-06-30  0.01062  0.02555 0.017794  0.00903
>
```

Portfolio Values at the end of June 2020

```
> port[c(1:3,nrow(port)),]
         date       date AMGN.Close AMGN.Adjusted JNJ.Close JNJ.Adjusted CSCO.Close CSCO.Adjusted MSFT.Close MSFT.Adjusted CVX.Close CVX.Adjusted
2019-07-02    1 2019-07-02        187           182       140          136       55.8          53.7        137           135      122.9        117.4
2019-07-03    2 2019-07-03        189           183       142          138       56.5          54.7        137           136      123.3        117.8
2019-07-05    3 2019-07-05        186           181       141          137       56.6          54.8        137           135      123.5        118.0
2020-06-30  252 2020-06-30        236           236       141          141       46.6          46.3        204           204       89.2         89.2
           XOM.Close XOM.Adjusted AMGN.ret JNJ.ret CSCO.ret MSFT.ret CVX.ret  XOM.ret
2019-07-02      75.7         71.5       NA      NA       NA       NA      NA       NA
2019-07-03      76.4         72.1   0.0104  0.0151  0.01839  0.00644 0.00342  0.00951
2019-07-05      76.1         71.9  -0.0140 -0.0110  0.00212 -0.00291 0.00154 -0.00406
2020-06-30      44.7         44.7   0.0186  0.0114  0.01062  0.02555 0.01779  0.00903
```

Equal-weighted Portfolio

Step 1: Rename the return variables

```
>    port[c(1:3,nrow(port)),]
                date AMGN.ret  JNJ.ret CSCO.ret MSFT.ret  CVX.ret  XOM.ret
2019-07-03 2019-07-03   0.0104  0.01507  0.01839  0.00644 0.003417  0.00951
2019-07-05 2019-07-05  -0.0140 -0.01105  0.00212 -0.00291 0.001540 -0.00406
2019-07-08 2019-07-08  -0.0199  0.00285 -0.00724 -0.00073 0.000486  0.00460
2020-06-30 2020-06-30   0.0186  0.01144  0.01062  0.02555 0.017794  0.00903
>
>    ewport <- port
>
>      ewport[c(1:3,nrow(ewport)),]
                date AMGN.ret  JNJ.ret CSCO.ret MSFT.ret  CVX.ret  XOM.ret
2019-07-03 2019-07-03   0.0104  0.01507  0.01839  0.00644 0.003417  0.00951
2019-07-05 2019-07-05  -0.0140 -0.01105  0.00212 -0.00291 0.001540 -0.00406
2019-07-08 2019-07-08  -0.0199  0.00285 -0.00724 -0.00073 0.000486  0.00460
2020-06-30 2020-06-30   0.0186  0.01144  0.01062  0.02555 0.017794  0.00903
>      names(ewport) <-
+      paste(c("date","AMGN","JNJ","CSCO","MSFT","CVX","XOM"))
>      rownames(ewport) <- seq(1:nrow(ewport))
>      ewport[c(1:3,nrow(ewport)),]
          date     AMGN      JNJ     CSCO     MSFT      CVX      XOM
1   2019-07-03   0.0104  0.01507  0.01839  0.00644 0.003417  0.00951
2   2019-07-05  -0.0140 -0.01105  0.00212 -0.00291 0.001540 -0.00406
3   2019-07-08  -0.0199  0.00285 -0.00724 -0.00073 0.000486  0.00460
251 2020-06-30   0.0186  0.01144  0.01062  0.02555 0.017794  0.00903
>
```

Step 2: Convert the data to gross returns

```
> # Convert the data to gross returns, by adding one to each
> # security's returns.
> # To reduce the number of variables overwrite the net return
> # series
>       ewport$AMGN <- 1+ewport$AMGN
>       ewport$JNJ <- 1+ewport$JNJ
>       ewport$CSCO <- 1+ewport$CSCO
>       ewport$MSFT <- 1+ewport$MSFT
>       ewport$CVX <- 1+ewport$CVX
>       ewport$XOM <- 1+ewport$XOM
>
>
>
>       ewport[c(1:3,nrow(ewport)),]
          date  AMGN   JNJ  CSCO  MSFT  CVX   XOM
1   2019-07-03 1.010 1.015 1.018 1.006 1.00 1.010
2   2019-07-05 0.986 0.989 1.002 0.997 1.00 0.996
3   2019-07-08 0.980 1.003 0.993 0.999 1.00 1.005
251 2020-06-30 1.019 1.011 1.011 1.026 1.02 1.009
> |
```

Step 3: Calculate the EW portfolio values for 2019 Q3

```
>
> # Calculate EW portfolio values for 3Q 2019
> ew.q3 <- subset(ewport,
+                 ewport$date >= as.Date("2019-07-01") &
+                   ewport$date <= as.Date("2019-09-30"))
> ew.q3[c(1:3,nrow(ew.q3)),]
         date  AMGN   JNJ  CSCO  MSFT CVX   XOM
1  2019-07-03 1.010 1.015 1.018 1.006   1 1.010
2  2019-07-05 0.986 0.989 1.002 0.997   1 0.996
3  2019-07-08 0.980 1.003 0.993 0.999   1 1.005
62 2019-09-30 0.993 1.006 1.012 1.009   1 0.988
> |
```

Step 4: Calculate the portfolio values for 2019 Q1

```
> # Calculate EW portfolio values for 1Q 2019 (cont.)
> # The return on Dec. 31, 2006 is not used (it is used as a
> #                                          placeholder used later for indexing for each security at
 $1)
> # Overwrite return on Dec. 31, 2019 (for each security) to 1.00
> ew.q3[1,2:7]<-1
> ew.q3[c(1:3,nrow(ew.q3)),]
         date  AMGN   JNJ  CSCO  MSFT CVX   XOM
1  2019-07-03 1.000 1.000 1.000 1.000   1 1.000
2  2019-07-05 0.986 0.989 1.002 0.997   1 0.996
3  2019-07-08 0.980 1.003 0.993 0.999   1 1.005
62 2019-09-30 0.993 1.006 1.012 1.009   1 0.988
> ew.q3$AMGN <- cumprod(ew.q3$AMGN)
> ew.q3$JNJ <- cumprod(ew.q3$JNJ)
> ew.q3$CSCO <- cumprod(ew.q3$CSCO)
> ew.q3$MSFT <- cumprod(ew.q3$MSFT)
> ew.q3$CVX <- cumprod(ew.q3$CVX)
> ew.q3$XOM <- cumprod(ew.q3$XOM)
>
>
>
> ew.q3[c(1:3,nrow(ew.q3)),]
         date  AMGN   JNJ  CSCO  MSFT   CVX   XOM
1  2019-07-03 1.000 1.000 1.000 1.000 1.000 1.000
2  2019-07-05 0.986 0.989 1.002 0.997 1.002 0.996
3  2019-07-08 0.966 0.992 0.995 0.996 1.002 1.001
62 2019-09-30 1.033 0.917 0.875 1.015 0.971 0.935
> |
```

Step 5: Calculate EW portfolio values for 2020 Q1

```
>
> ew.q3[c(1:3,nrow(ew.q3)),]
         date  AMGN   JNJ  CSCO  MSFT   CVX   XOM AMGN.idx CSCO.idx JNJ.idx MSFT.idx XOM.idx
1  2019-07-03 1.000 1.000 1.000 1.000 1.000 1.000    0.167    0.167   0.167    0.167   0.167
2  2019-07-05 0.986 0.989 1.002 0.997 1.002 0.996    0.164    0.167   0.165    0.166   0.166
3  2019-07-08 0.966 0.992 0.995 0.996 1.002 1.001    0.161    0.166   0.165    0.166   0.167
62 2019-09-30 1.033 0.917 0.875 1.015 0.971 0.935    0.172    0.146   0.153    0.169   0.156
   CVX.idx
1    0.167
2    0.167
3    0.167
62   0.162
>
```

Step 6: Calculate the index value for each security during the quarter

```
> # Calculate the index value for each security during the quarter
> # The value of a portfolio is equal to the value of the components.
> # To calculate the value of the EW portfolio on each day, sum the
> # values of the three index variables.
>     q3.val <- data.frame(rowSums(ew.q3[,8:13]))
>     q3.val[c(1:3,nrow(q3.val)),]
[1] 1.000 0.995 0.992 0.958
>     names(q3.val) <- paste("port.val")
>     q3.val$date <- ew.q3$date
>     q3.val[c(1:3,nrow(q3.val)),]
   port.val       date
1     1.000 2019-07-03
2     0.995 2019-07-05
3     0.992 2019-07-08
62    0.958 2019-09-30
>
```

Step 7: Calculate the index value for each security during the quarter

```
> # Calculate the index value for each security during the quarter
> # q3.val holds the portfolio values for the third quarter
> # q4.inv holds the aggregate portfolio value at the end of Q3
> # q4.inv will be redistributed equally among six stocks at the
> # beginning of the fourth quarter
> # The value of 1.9156 is used at the beginning of the second quarter
> # instead of 1.00
>       q4.inv <- q3.val[nrow(q3.val),1]
>       q4.inv
[1] 0.958
>
```

Similarly, we construct the equal-weighted portfolio for the 2019 Q4, 2020 Q1 and 2020 Q2 and then combine them into a single data object as below:

```
> # Combine quarterly EW portfolio values into one data object
> # (ew.portval)
> # Please note that this calculations do not include transactions
> # costs incurred during each Q rebalancing.
>     ew.portval <- rbind(q3.val,q4.val,q1.val,q2.val)
>     ew.portval[c(1:3,nrow(ew.portval)),]
    port.val       date
1      1.000 2019-07-03
2      0.995 2019-07-05
3      0.992 2019-07-08
251    1.199 2020-06-30
>
>
>     ew.portval
    port.val       date
1      1.000 2019-07-03
2      0.995 2019-07-05
3      0.992 2019-07-08
4      0.991 2019-07-09
5      1.001 2019-07-10
6      0.997 2019-07-11
7      0.991 2019-07-12
8      0.991 2019-07-15
9      0.983 2019-07-16
10     0.977 2019-07-17
11     0.981 2019-07-18
12     0.978 2019-07-19
13     0.977 2019-07-22
14     0.980 2019-07-23
15     0.982 2019-07-24
16     0.978 2019-07-25
17     0.977 2019-07-26
18     0.983 2019-07-29
19     0.980 2019-07-30
20     0.975 2019-07-31
21     0.969 2019-08-01
```

Portfolio value as of June 30 2020:

```
    port.val       date
189  0.97203 2020-04-01
190  1.02780 2020-04-02
191  1.01402 2020-04-03
251  1.19884 2020-06-30
```

Recommendation/Conclusion:

The portfolio gain js $0.19884 as of June 30 2020.

The portfolio value as per each quarter is as follows;

Quarter	Value of Portfolio at end of quarter
Q3 - 2019 July - Sep 2019	0.95778
Q4 - 2019 Oct - Dec 2019	1.0886
Q1 - 2020 Jan - March 2020	0.79006
Q2 - 2020 April - June 2020	1.1988

As we can see, the EW portfolio value fluctuated rapidly in the 2019-2020 cycle. The worst value was $0.79006 at the end of Q1-2020 which denotes an approximately 21% loss in the value of the portfolio. However, stock prices rapidly increased in the Q2 -2020 and the portfolio is profitable again at 1.1988. which is a Y-O-Y profit of 20%.

Value-weighted Portfolio

Model Description

Step 1: Create a data object with only relevant data

```
>       vport <- data.AMGN[,c(4,5)]
>       vport <- merge(vport, data.JNJ[,c(4,5)])
>
>       vport <- merge(vport,data.CSCO[,c(4,5)])
>       vport <- merge(vport,data.MSFT[,c(4,5)])
>
>       vport <- merge(vport,data.CVX[,c(4,5)])
>       vport <- merge(vport,data.XOM[,c(4,5)])
>
> vport[c(1:3,nrow(vport)),]
           AMGN.Close AMGN.Adjusted JNJ.Close JNJ.Adjusted CSCO.Close CSCO.Adjusted MSFT.Close
2019-07-02        187           182       140          136       55.8          53.7        137
2019-07-03        189           183       142          138       56.5          54.7        137
2019-07-05        186           181       141          137       56.6          54.8        137
2020-06-30        236           236       141          141       46.6          46.3        204
           MSFT.Adjusted CVX.Close CVX.Adjusted XOM.Close XOM.Adjusted
2019-07-02           135     122.9        117.4      75.7         71.5
2019-07-03           136     123.3        117.8      76.4         72.1
2019-07-05           135     123.5        118.0      76.1         71.9
2020-06-30           204      89.2         89.2      44.7         44.7
```

Step 2: Calculate total returns of each security using Adjusted Close prices

```
>     vport$AMGN.ret <- Delt(vport$AMGN.Adjusted)
>       vport$JNJ.ret <- Delt(vport$JNJ.Adjusted)
>       vport$CSCO.ret <- Delt(vport$CSCO.Adjusted)
>       vport$MSFT.ret <- Delt(vport$MSFT.Adjusted)
>       vport$CVX.ret <- Delt(vport$CVX.Adjusted)
>       vport$XOM.ret <- Delt(vport$XOM.Adjusted)
>
>
> vport[c(1:3,nrow(vport)),]
           AMGN.Close AMGN.Adjusted JNJ.Close JNJ.Adjusted CSCO.Close CSCO.Adjusted MSFT.Close
2019-07-02        187           182       140          136       55.8          53.7        137
2019-07-03        189           183       142          138       56.5          54.7        137
2019-07-05        186           181       141          137       56.6          54.8        137
2020-06-30        236           236       141          141       46.6          46.3        204
           MSFT.Adjusted CVX.Close CVX.Adjusted XOM.Close XOM.Adjusted AMGN.ret JNJ.ret CSCO.ret
2019-07-02           135     122.9        117.4      75.7         71.5       NA      NA       NA
2019-07-03           136     123.3        117.8      76.4         72.1   0.0104  0.0151  0.01839
2019-07-05           135     123.5        118.0      76.1         71.9  -0.0140 -0.0110  0.00212
2020-06-30           204      89.2         89.2      44.7         44.7   0.0186  0.0114  0.01062
           MSFT.ret CVX.ret  XOM.ret
2019-07-02       NA      NA       NA
2019-07-03  0.00644 0.00342  0.00951
2019-07-05 -0.00291 0.00154 -0.00406
2020-06-30  0.02555 0.01779  0.00903
```

Step 3: Bind the date and total returns of each security using Adjusted closing prices.

```
> vport <- cbind(data.frame(index(vport)),
+                data.frame(vport))
> names(vport)[1] <- paste("date")
> vport[c(1:3,nrow(vport)),]
                date AMGN.Close AMGN.Adjusted JNJ.Close JNJ.Adjusted CSCO.Close CSCO.Adjusted
2019-07-02 2019-07-02        187           182       140          136       55.8          53.7
2019-07-03 2019-07-03        189           183       142          138       56.5          54.7
2019-07-05 2019-07-05        186           181       141          137       56.6          54.8
2020-06-30 2020-06-30        236           236       141          141       46.6          46.3
           MSFT.Close MSFT.Adjusted CVX.Close CVX.Adjusted XOM.Close XOM.Adjusted AMGN.ret JNJ.ret
2019-07-02        137           135     122.9        117.4      75.7         71.5       NA      NA
2019-07-03        137           136     123.3        117.8      76.4         72.1   0.0104  0.0151
2019-07-05        137           135     123.5        118.0      76.1         71.9  -0.0140 -0.0110
2020-06-30        204           204      89.2         89.2      44.7         44.7   0.0186  0.0114
           CSCO.ret MSFT.ret CVX.ret  XOM.ret
2019-07-02       NA       NA      NA       NA
2019-07-03  0.01839  0.00644 0.00342  0.00951
2019-07-05  0.00212 -0.00291 0.00154 -0.00406
2020-06-30  0.01062  0.02555 0.01779  0.00903
> |
```

Step 4: Calculate the market capitalization of each security in the portfolio

```
> date <- seq(as.Date("2019-07-1"),as.Date("2020-06-30"),by=1)
> date <- data.frame(date)
> date[c(1:3,nrow(date)),]
[1] "2019-07-01" "2019-07-02" "2019-07-03" "2020-06-30"
>
>     vwport <- vport
>     vwport[c(1:3,nrow(vwport)),]
                date AMGN.Close AMGN.Adjusted JNJ.Close JNJ.Adjusted CSCO.Close CSCO.Adjusted
2019-07-02 2019-07-02        187           182       140          136       55.8          53.7
2019-07-03 2019-07-03        189           183       142          138       56.5          54.7
2019-07-05 2019-07-05        186           181       141          137       56.6          54.8
2020-06-30 2020-06-30        236           236       141          141       46.6          46.3
           MSFT.Close MSFT.Adjusted CVX.Close CVX.Adjusted XOM.Close XOM.Adjusted AMGN.ret JNJ.ret
2019-07-02        137           135     122.9        117.4      75.7         71.5       NA      NA
2019-07-03        137           136     123.3        117.8      76.4         72.1   0.0104  0.0151
2019-07-05        137           135     123.5        118.0      76.1         71.9  -0.0140 -0.0110
2020-06-30        204           204      89.2         89.2      44.7         44.7   0.0186  0.0114
           CSCO.ret MSFT.ret CVX.ret  XOM.ret
2019-07-02       NA       NA      NA       NA
2019-07-03  0.01839  0.00644 0.00342  0.00951
2019-07-05  0.00212 -0.00291 0.00154 -0.00406
2020-06-30  0.01062  0.02555 0.01779  0.00903
> |
```

Step 5: Index of viewport can be changed to an indicator of observation number

```
>     rownames(vwport) <- seq(1:nrow(vwport))
>     vwport[c(1:3,nrow(vwport)),]
          date AMGN.Close AMGN.Adjusted JNJ.Close JNJ.Adjusted CSCO.Close CSCO.Adjusted MSFT.Close
1   2019-07-02        187           182       140          136       55.8          53.7        137
2   2019-07-03        189           183       142          138       56.5          54.7        137
3   2019-07-05        186           181       141          137       56.6          54.8        137
252 2020-06-30        236           236       141          141       46.6          46.3        204
    MSFT.Adjusted CVX.Close CVX.Adjusted XOM.Close XOM.Adjusted AMGN.ret JNJ.ret CSCO.ret MSFT.ret
1             135     122.9        117.4      75.7         71.5       NA      NA       NA       NA
2             136     123.3        117.8      76.4         72.1   0.0104  0.0151  0.01839  0.00644
3             135     123.5        118.0      76.1         71.9  -0.0140 -0.0110  0.00212 -0.00291
252           204      89.2         89.2      44.7         44.7   0.0186  0.0114  0.01062  0.02555
    CVX.ret  XOM.ret
1        NA       NA
2   0.00342  0.00951
3   0.00154 -0.00406
252 0.01779  0.00903
> |
```

Step 6: Conversion of net returns to gross returns

```
         date AMGN.Close AMGN.Adjusted JNJ.Close JNJ.Adjusted CSCO.Close CSCO.Adjusted MSFT.Close
1   2019-07-02        187           182       140          136       55.8          53.7        137
2   2019-07-03        189           183       142          138       56.5          54.7        137
3   2019-07-05        186           181       141          137       56.6          54.8        137
252 2020-06-30        236           236       141          141       46.6          46.3        204
    MSFT.Adjusted CVX.Close CVX.Adjusted XOM.Close XOM.Adjusted AMGN.ret JNJ.ret CSCO.ret MSFT.ret
1             135     122.9        117.4      75.7         71.5       NA      NA       NA       NA
2             136     123.3        117.8      76.4         72.1    1.010   1.015     1.02    1.006
3             135     123.5        118.0      76.1         71.9    0.986   0.989     1.00    0.997
252           204      89.2         89.2      44.7         44.7    1.019   1.011     1.01    1.026
    CVX.ret XOM.ret
1        NA      NA
2      1.00   1.010
3      1.00   0.996
252    1.02   1.009
```

Step 7: Obtain shares outstanding data from SEC filings (EDGAR database Form 10k - Form 10-Q). Enter the data manually. Values are in units of billion USD.

```
> PRICE.qtr$AMGN.shout <- c(0.6021, 0.5962, 0.5914, 0.588)
>
>     PRICE.qtr$JNJ.shout <- c(2.642, 2.631, 2.633, 2.632)
>
>     PRICE.qtr$CSCO.shout <- c(4.25, 4.244, 4.242, 4.222)
>
>     PRICE.qtr$MSFT.shout <-c(7.643, 7.634, 7.611, 7.59)
>
>     PRICE.qtr$CVX.shout <-c(1.898, 1.891, 1.882, 1.867)
>
>     PRICE.qtr$XOM.shout <-  c(4.231, 4.231, 4.234, 4.228)
>
```

Step 8: Calculation of market capitalization of each security in the portfolio:

```
>     weights$tot.mcap <- rowSums(weights[14:19])
>     weights
        date AMGN.Close JNJ.Close CSCO.Close MSFT.Close CVX.Close XOM.Close AMGN.shout JNJ.shout
1 2019-07-02        187       140        55.8        137     122.9      75.7       0.602      2.64
2 2019-10-01        193       130        47.7        137     116.0      68.9       0.596      2.63
3 2020-01-02        240       146        48.4        161     121.4      70.9       0.591      2.63
4 2020-04-01        198       129        38.3        152      68.6      37.5       0.588      2.63
  CSCO.shout MSFT.shout CVX.shout XOM.shout AMGN.mcap CSCO.mcap JNJ.mcap MSFT.mcap CVX.mcap
1       4.25       7.64      1.90      4.23       112       237      370      1044      233
2       4.24       7.63      1.89      4.23       115       203      342      1046      219
3       4.24       7.61      1.88      4.23       142       205      384      1222      229
4       4.22       7.59      1.87      4.23       116       162      339      1155      128
  XOM.mcap tot.mcap AMGN.wgt CSCO.wgt JNJ.wgt MSFT.wgt CVX.wgt XOM.wgt
1      320     2317   0.0485   0.1024   0.160    0.451  0.1007  0.1383
2      292     2217   0.0519   0.0914   0.154    0.472  0.0989  0.1316
3      300     2483   0.0572   0.0827   0.155    0.492  0.0920  0.1209
4      159     2058   0.0565   0.0786   0.165    0.561  0.0622  0.0771
>
```

Step 9: Calculate the quarter-end weights of each security in the portfolio.

```
> weights$AMGN.wgt <-weights$AMGN.mcap/weights$tot.mcap
>       weights$CSCO.wgt <-weights$CSCO.mcap/weights$tot.mcap
>       weights$JNJ.wgt <-weights$JNJ.mcap/weights$tot.mcap
>       weights$MSFT.wgt <-weights$MSFT.mcap/weights$tot.mcap
>       weights$CVX.wgt <-weights$CVX.mcap/weights$tot.mcap
>       weights$XOM.wgt <-weights$XOM.mcap/weights$tot.mcap
>       weights
        date AMGN.Close JNJ.Close CSCO.Close MSFT.Close CVX.Close XOM.Close AMGN.shout JNJ.shout
1 2019-07-02        187       140        55.8        137     122.9      75.7      0.602      2.64
2 2019-10-01        193       130        47.7        137     116.0      68.9      0.596      2.63
3 2020-01-02        240       146        48.4        161     121.4      70.9      0.591      2.63
4 2020-04-01        198       129        38.3        152      68.6      37.5      0.588      2.63
  CSCO.shout MSFT.shout CVX.shout XOM.shout AMGN.mcap CSCO.mcap JNJ.mcap MSFT.mcap CVX.mcap
1       4.25       7.64      1.90      4.23       112       237      370      1044       233
2       4.24       7.63      1.89      4.23       115       203      342      1046       219
3       4.24       7.61      1.88      4.23       142       205      384      1222       229
4       4.22       7.59      1.87      4.23       116       162      339      1155       128
  XOM.mcap tot.mcap AMGN.wgt CSCO.wgt JNJ.wgt MSFT.wgt CVX.wgt XOM.wgt
1      320     2317   0.0485   0.1024   0.160    0.451  0.1007  0.1383
2      292     2217   0.0519   0.0914   0.154    0.472  0.0989  0.1316
3      300     2483   0.0572   0.0827   0.155    0.492  0.0920  0.1209
4      159     2058   0.0565   0.0786   0.165    0.561  0.0622  0.0771
```

Step 10: Calculate the quarter-end weights of each security in the portfolio.

```
> weights$AMGN.wgt <-weights$AMGN.mcap/weights$tot.mcap
>       weights$CSCO.wgt <-weights$CSCO.mcap/weights$tot.mcap
>       weights$JNJ.wgt <-weights$JNJ.mcap/weights$tot.mcap
>       weights$MSFT.wgt <-weights$MSFT.mcap/weights$tot.mcap
>       weights$CVX.wgt <-weights$CVX.mcap/weights$tot.mcap
>       weights$XOM.wgt <-weights$XOM.mcap/weights$tot.mcap
>       weights
        date AMGN.Close JNJ.Close CSCO.Close MSFT.Close CVX.Close XOM.Close AMGN.shout JNJ.shout
1 2019-07-02        187       140        55.8        137     122.9      75.7      0.602      2.64
2 2019-10-01        193       130        47.7        137     116.0      68.9      0.596      2.63
3 2020-01-02        240       146        48.4        161     121.4      70.9      0.591      2.63
4 2020-04-01        198       129        38.3        152      68.6      37.5      0.588      2.63
  CSCO.shout MSFT.shout CVX.shout XOM.shout AMGN.mcap CSCO.mcap JNJ.mcap MSFT.mcap CVX.mcap
1       4.25       7.64      1.90      4.23       112       237      370      1044       233
2       4.24       7.63      1.89      4.23       115       203      342      1046       219
3       4.24       7.61      1.88      4.23       142       205      384      1222       229
4       4.22       7.59      1.87      4.23       116       162      339      1155       128
  XOM.mcap tot.mcap AMGN.wgt CSCO.wgt JNJ.wgt MSFT.wgt CVX.wgt XOM.wgt
1      320     2317   0.0485   0.1024   0.160    0.451  0.1007  0.1383
2      292     2217   0.0519   0.0914   0.154    0.472  0.0989  0.1316
3      300     2483   0.0572   0.0827   0.155    0.492  0.0920  0.1209
4      159     2058   0.0565   0.0786   0.165    0.561  0.0622  0.0771
>
```

Step 11: Merge weight into date and calculate quarterly vw portfolio values:

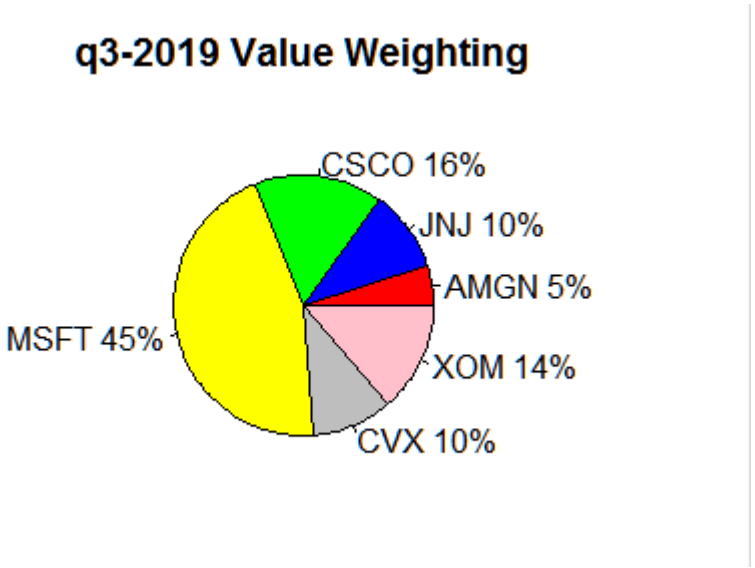
```
> #Merge weight into date
> vwret <- na.locf(merge(date,WEIGHT,by="date",all.x=TRUE))
> vwret[c(1:3,nrow(vwret)),]
          date AMGN.wgt CSCO.wgt JNJ.wgt MSFT.wgt CVX.wgt XOM.wgt
3   2019-07-03   0.0485   0.1024   0.160    0.451  0.1007  0.1383
4   2019-07-04   0.0485   0.1024   0.160    0.451  0.1007  0.1383
5   2019-07-05   0.0485   0.1024   0.160    0.451  0.1007  0.1383
366 2020-06-30   0.0565   0.0786   0.165    0.561  0.0622  0.0771
>
>
> # Calculating the quarterly Vw portfolio values (similar to EW
> #                                                 portfolio)
> # Check date as.date and wgt as.num
> str(vwret)
'data.frame':   364 obs. of  7 variables:
 $ date    : Date, format: "2019-07-03" "2019-07-04" "2019-07-05" ...
 $ AMGN.wgt: num  0.0485 0.0485 0.0485 0.0485 0.0485 ...
 $ CSCO.wgt: num  0.102 0.102 0.102 0.102 0.102 ...
 $ JNJ.wgt : num  0.16 0.16 0.16 0.16 0.16 ...
 $ MSFT.wgt: num  0.451 0.451 0.451 0.451 0.451 ...
 $ CVX.wgt : num  0.101 0.101 0.101 0.101 0.101 ...
 $ XOM.wgt : num  0.138 0.138 0.138 0.138 0.138 ...
 - attr(*, "na.action")= 'omit' Named int [1:2] 1 2
  ..- attr(*, "names")= chr [1:2] "1" "2"
>
```

Step 12: Calculate quarterly vw portfolio values:

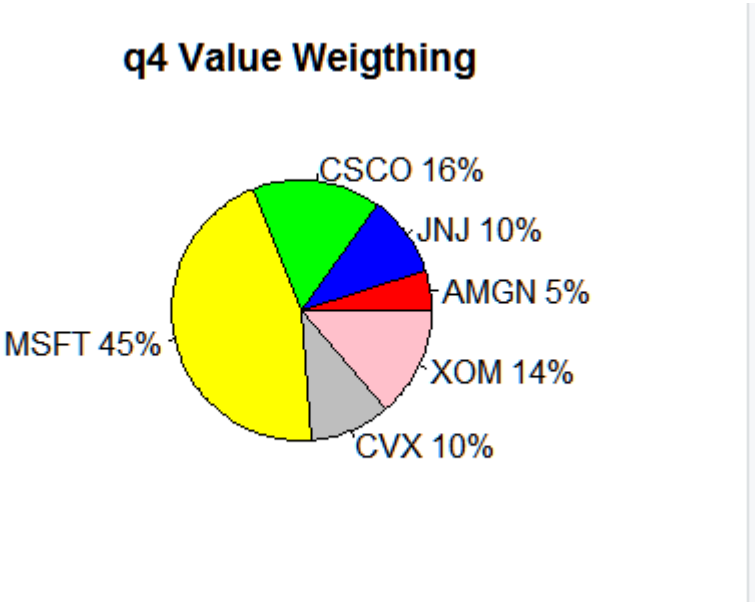
```
> # Calculating the quarterly Vw portfolio values (similar to EW
> #                                                 portfolio)
> # Check date as.date and wgt as.num
> str(vwret)
'data.frame':   364 obs. of  7 variables:
 $ date    : Date, format: "2019-07-03" "2019-07-04" "2019-07-05" ...
 $ AMGN.wgt: num  0.0485 0.0485 0.0485 0.0485 0.0485 ...
 $ CSCO.wgt: num  0.102 0.102 0.102 0.102 0.102 ...
 $ JNJ.wgt : num  0.16 0.16 0.16 0.16 0.16 ...
 $ MSFT.wgt: num  0.451 0.451 0.451 0.451 0.451 ...
 $ CVX.wgt : num  0.101 0.101 0.101 0.101 0.101 ...
 $ XOM.wgt : num  0.138 0.138 0.138 0.138 0.138 ...
 - attr(*, "na.action")= 'omit' Named int [1:2] 1 2
  ..- attr(*, "names")= chr [1:2] "1" "2"
> q3.vw.wgt <-  subset(vwret,vwret$date == as.Date("2019-07-03"))
> q3.vw.wgt
        date AMGN.wgt CSCO.wgt JNJ.wgt MSFT.wgt CVX.wgt XOM.wgt
3 2019-07-03   0.0485    0.102    0.16    0.451   0.101   0.138
>
> q4.vw.wgt <-  subset(vwret,vwret$date==as.Date("2019-10-01"))
> q4.vw.wgt
         date AMGN.wgt CSCO.wgt JNJ.wgt MSFT.wgt CVX.wgt XOM.wgt
93 2019-10-01   0.0485    0.102    0.16    0.451   0.101   0.138
>
> q1.vw.wgt <-
+   subset(vwret,vwret$date==as.Date("2020-01-01"))
> q1.vw.wgt
          date AMGN.wgt CSCO.wgt JNJ.wgt MSFT.wgt CVX.wgt XOM.wgt
185 2020-01-01   0.0519   0.0914   0.154    0.472  0.0989   0.132
>
> q2.vw.wgt <-
+   subset(vwret,vwret$date==as.Date("2020-04-01"))
> q2.vw.wgt
          date AMGN.wgt CSCO.wgt JNJ.wgt MSFT.wgt CVX.wgt XOM.wgt
276 2020-04-01   0.0572   0.0827   0.155    0.492   0.092   0.121
>
```

Output Analysis: Creation of pie charts

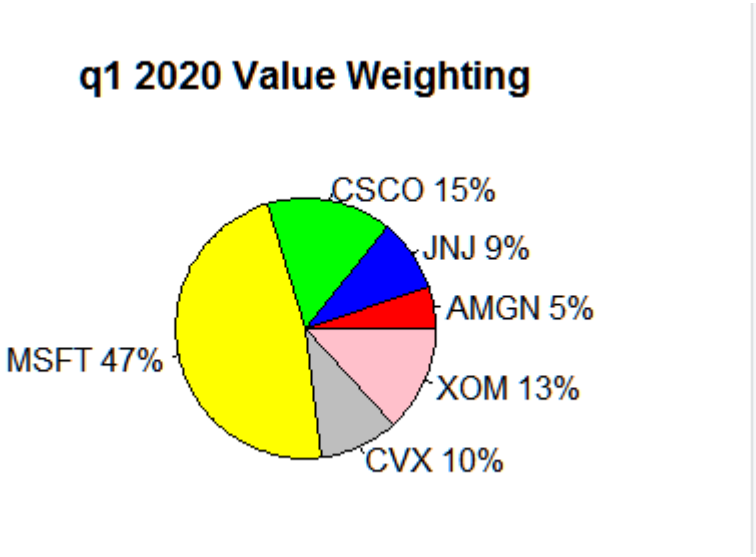
2019 Q3



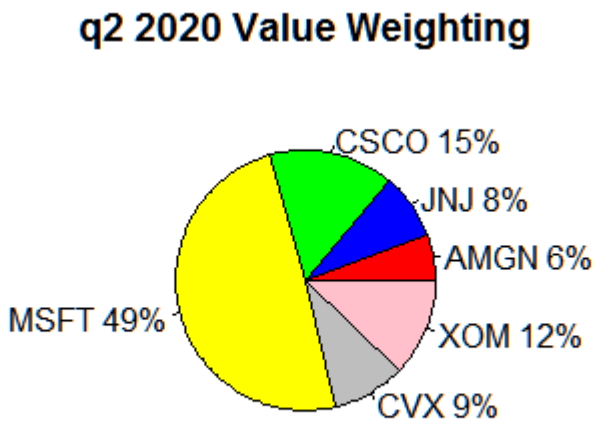
2019 Q4



2020 Q1



2020 Q2



Value of $1 investment in equal weighted and value weighted portfolio of AMGN, JNJ, CSCO, MSFT, CVX, XOM

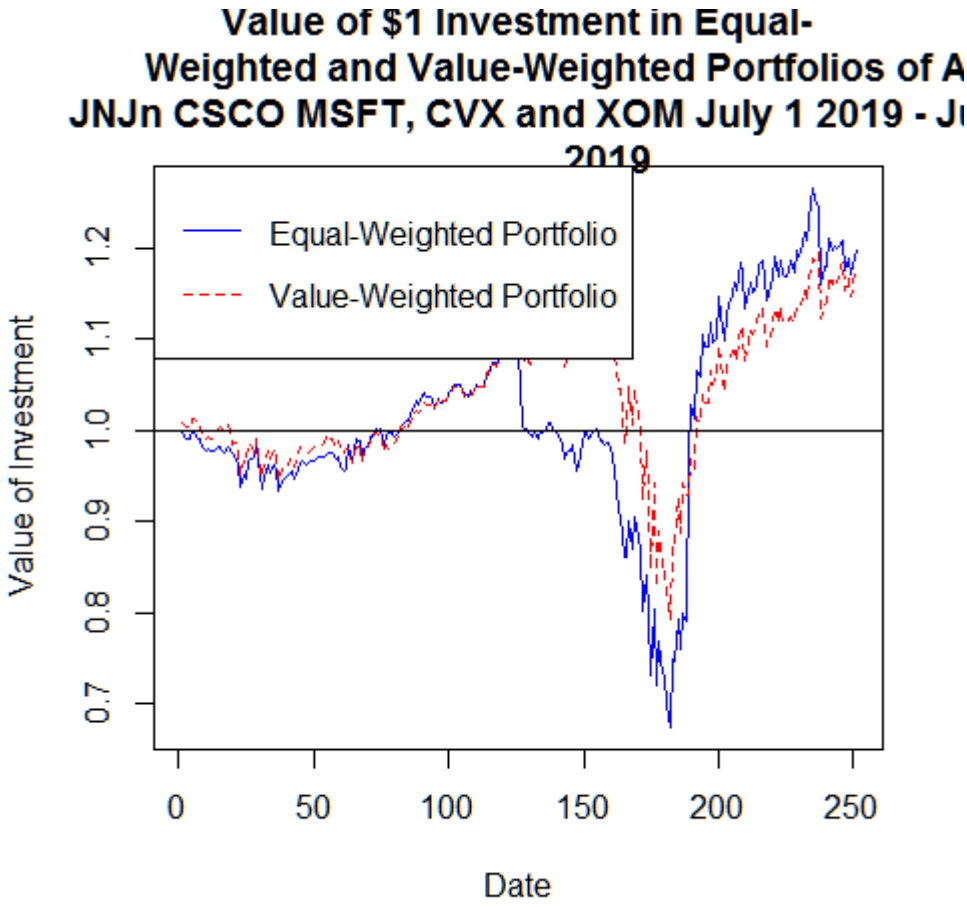
Conclusions/Recommendation:

Value Weighted Portfolio value per quarter:

Quarter	Value of Portfolio at end of quarter
Q3 - 2019 July - Sep 2019	0.98196
Q4 - 2019 Oct - Dec 2019	1.0892

Q1 - 2020 Jan - March 2020	0.92979
Q2 - 2020 April - June 2020	1.1803

As we can see from the above table, the VW portfolio was conservative and did not sharply fall as the EW Portfolio in the Q1 2020. The gain from VW is still impressive and comparable to EW with 1.1803 at the end of June 2020 which amounts to around 18% growth in the year 2019-2020.



Conclusions/Recommendations:

In the above plot, we can see that $1 investment in an equal weighted and value weighted portfolio shows a steady run and then a dip around March 2020 and then a steady bounce back after March 2020, primarily due to Microsoft performance and high weights attributed to Microsoft.

Risk Assessment and Hyperbolic Distributions:

Model Description:

We performed a risk assessment for each of the 3 sectors in our portfolio. The following methods have been used for the purpose of calculating the risk assessment:

Mathematical matrix calculations:

Create a vector with Equal weights for the portfolio. Then create a matrix for the returns:

```
> #Create a vector of weights - All 16.667%
>
>               WGT.2asset <- c(0.1666666666666667,0.1666666666666667,0.16666666666666€
>               WGT.2asset <- matrix(WGT.2asset,1)
>               WGT.2asset
        [,1]    [,2]    [,3]    [,4]    [,5]    [,6]
[1,] 0.16667 0.16667 0.16667 0.16667 0.16667 0.16667
> tWGT.2asset <- t(WGT.2asset)
> tWGT.2asset
        [,1]
[1,] 0.16667
[2,] 0.16667
[3,] 0.16667
[4,] 0.16667
[5,] 0.16667
[6,] 0.16667
```

Calculate the Covariance of the matrix as below.

```
>              cov(mat.Ret)
            AMGN.Ret    JNJ.Ret   CSCO.ret   MSFT.ret    CVX.ret    XOM.ret
AMGN.Ret 0.00051849 0.00030944 0.00038797 0.00040466 0.00039969 0.00033257
JNJ.Ret  0.00030944 0.00037596 0.00033897 0.00034130 0.00037464 0.00028291
CSCO.ret 0.00038797 0.00033897 0.00069434 0.00051682 0.00056076 0.00047454
MSFT.ret 0.00040466 0.00034130 0.00051682 0.00064480 0.00059987 0.00046719
CVX.ret  0.00039969 0.00037464 0.00056076 0.00059987 0.00124650 0.00087452
XOM.ret  0.00033257 0.00028291 0.00047454 0.00046719 0.00087452 0.00084547
```

In order to annualize the covariance, 252 days have been considered from 2019-2020, so we have multiplied by 252

```
>               VCOV.2asset <- cov(mat.Ret)*252
>               VCOV.2asset
          AMGN.Ret  JNJ.Ret CSCO.ret MSFT.ret CVX.ret  XOM.ret
AMGN.Ret 0.130658 0.077978 0.097768 0.101974 0.10072 0.083807
JNJ.Ret  0.077978 0.094743 0.085421 0.086008 0.09441 0.071293
CSCO.ret 0.097768 0.085421 0.174973 0.130240 0.14131 0.119585
MSFT.ret 0.101974 0.086008 0.130240 0.162491 0.15117 0.117733
CVX.ret  0.100723 0.094410 0.141311 0.151167 0.31412 0.220380
XOM.ret  0.083807 0.071293 0.119585 0.117733 0.22038 0.213059
```

Calculate the portfolio risk and Portfolio Standard Deviation.

```
> # Calculate the portfolio risk (remmeber %*% is used to multiply a matrix)
>
>
>             mat.var2asset <- WGT.2asset %*% VCOV.2asset %*% tWGT.2asset
>             mat.var2asset
         [,1]
[1,] 0.1236
>
>          # To calculate the portfolio stndard deviation, use the sqrt command.
>
>             mat.sd2asset <- sqrt(mat.var2asset)
>             mat.sd2asset
          [,1]
[1,] 0.35157
```

Suitable distributions for returns

Model description:

GHD, HYP and NIG are mathematical techniques for calculating the distribution for returns. We will be fitting the data using GHD, HYP and NIG commands. Then we will plot the combine density functions. Q-Q plot will help us understand the quantile distribution. 'Lik.ratio.test' will be used to compare the best model. We can then calculate and plot the VaR (using all models). Similarly, ES can be plotted for all models. For illustrative purposes, we have plotted the Hyperbolic distributions for Amgen (GHD, HYP and NIG). We have also plotted the QQ curve which describes how closely the data fits the model. In our project report, we will build models for all our sectors.

Following aspects have been covered as part of the portfolio distribution analysis:

Suitable distributions for returns

1. Fit the data using GHD, HYP and NIG

2. Plot the combine density functions

3. Create a Q-Q plot

4. Make a model recommendation using lik.ratio.test

5. Calculate and plot the VaR (using all models)

6. Calculate and plot the ES (using all models)

Model description:

We investigated the trajectory of the VaR and ES risk measures according to each of the models. The two risks (VaR, ES) measures are derived from the fitted GHD, HYP, and NIG distributions for the returns.

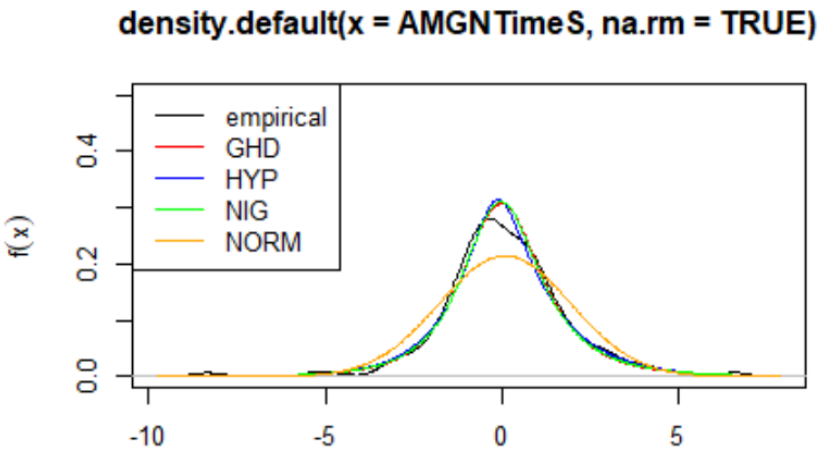
These measures are calculated for Confidence Interval ranging from 95.0% to the 99.9% levels. The resulting trajectories of the VaR and ES are then compared to their empirical counterparts.

For the **ES,** the mean of the lower quartile values is used. First, a sequence of probabilities is created for which the VaR and ES are to be computed. Because we are dealing with returns instead of losses, these are defined for the left tail of the distribution. VaR is computed by utilizing the quantile function for the GHD. As losses are expressed as positive numbers, the absolute values of the quantiles returned by the function are used.

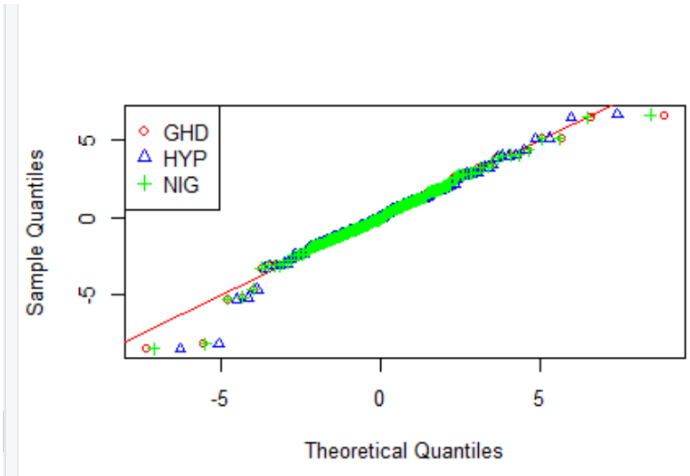
The VaR based on the normal distribution can be computed by providing the necessary estimates for the location and scale. The VaR values thus # determined are compared to their empirical counterparts, which are # determined by the quantile() function.

Output Analysis:

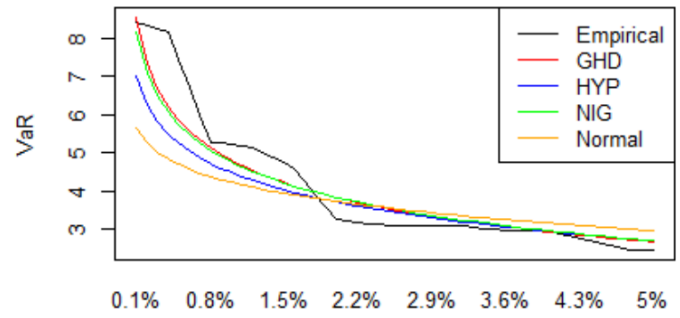
1 - Amgen Distribution, VaR and ES:



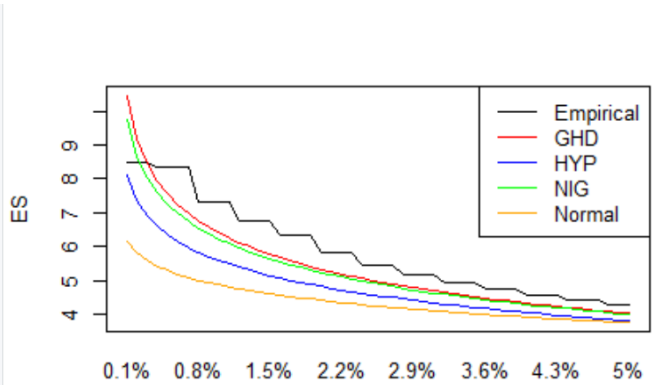
Quantile Plot:



VaR:



ES:



Conclusion/recommendation:

The above graph shows the VaR and Estimated shortfall for Amgen stocks using Empirical, GHD, HYP, NIG and normal. At 99.9% CI, the VaR distribution plot shows a high risk of 9% for the GHD and NIG model. As we increase the confidence interval to 95%, the risk proportion for all models goes down exponentially and converges to around 3.5%.

For ES, at 0.1% CI, the GHD and NIG models show a ES shortfall of 10% and at 95% the ES plot comes down to 4%.

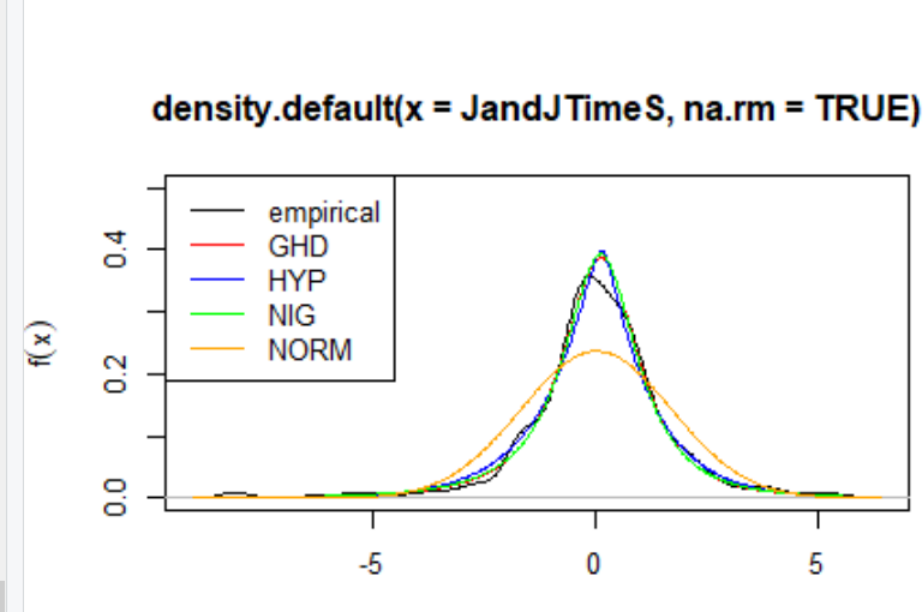
JNJ - AIC:

```
> head(AIC$fit.table)
   model symmetric   lambda alpha.bar       mu  sigma    gamma    aic     llh converged n.iter
8    NIG      TRUE -0.50000   0.38345 0.065589 1.6850  0.00000 912.07 -453.03      TRUE    134
10     t      TRUE -1.26977   0.00000 0.072061 2.0819  0.00000 913.01 -453.50      TRUE    136
3    NIG     FALSE -0.50000   0.39884 0.132576 1.6703 -0.13200 913.30 -452.65      TRUE    155
6   ghyp      TRUE -0.75841   0.33562 0.068295 1.7030  0.00000 913.91 -452.95      TRUE    265
5      t     FALSE -1.30026   0.00000 0.132030 2.0089 -0.16823 914.29 -453.15      TRUE    275
1   ghyp     FALSE -0.78192   0.34753 0.135257 1.6860 -0.13450 915.12 -452.56      TRUE    464
```

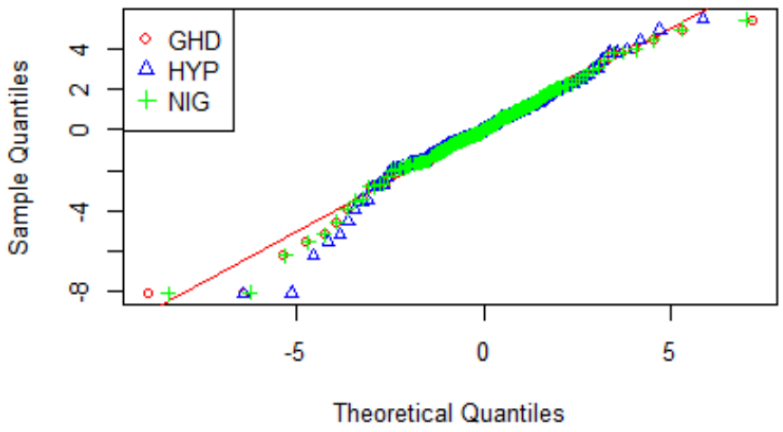
LIK Ratio Test:

```
> LRghdnig <- lik.ratio.test(ghdfit, nigfit)
> LRghdnig
$statistic
      L
0.91096

$p.value
[1] 0.66583

$df
[1] 1

$H0
[1] TRUE

> LRghdhyp <- lik.ratio.test(ghdfit, hypfit)
> LRghdhyp
$statistic
       L
0.037709

$p.value
[1] 0.010455

$df
[1] 1

$H0
[1] FALSE
```

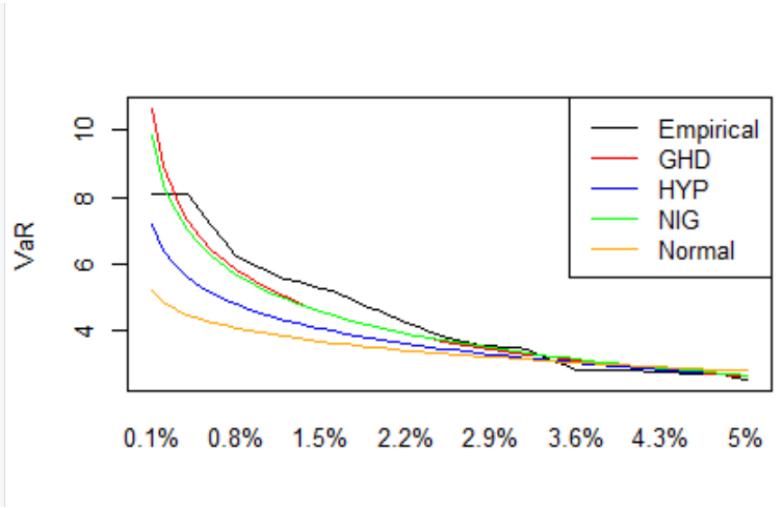
JNJ Density Plot:



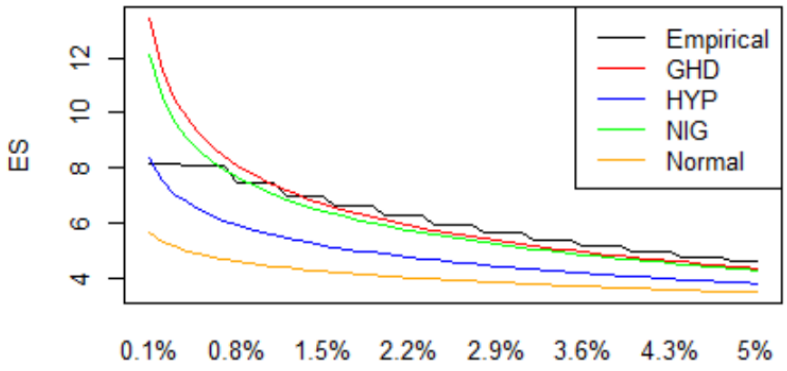
JNJ QQ Plot:



VAR for JnJ:



ES For JNJ:



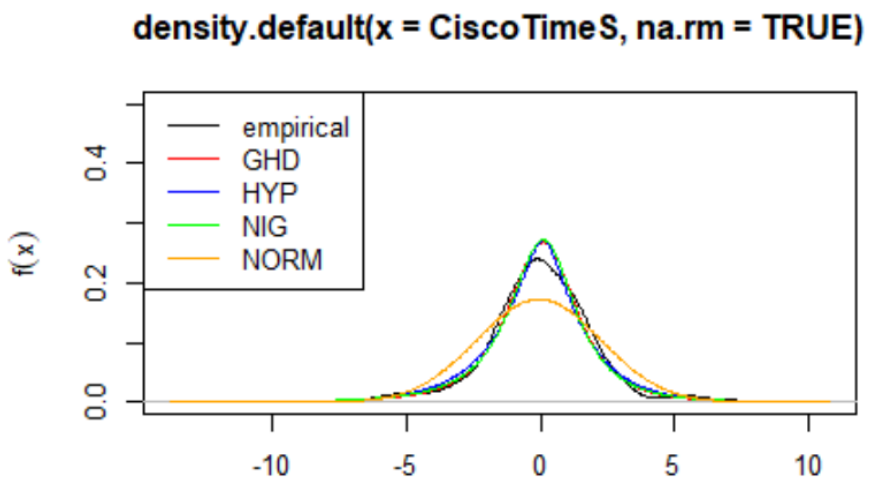
Cisco - AIC:

```
> summary(AIC$fit.table)
    model            symmetric          lambda         alpha.bar          mu                sigma          gamma              aic
 Length:11          Mode :logical   Min.   :-1.415   Min.   :0.000   Min.   :-0.06839   Min.   :2.17   Min.   :-0.2249   Min.   :1078
 Class :character   FALSE:5         1st Qu.:-1.073   1st Qu.:0.000   1st Qu.: 0.00274   1st Qu.:2.21   1st Qu.:-0.2009   1st Qu.:1079
 Mode  :character   TRUE :6         Median :-0.500   Median :0.305   Median : 0.02506   Median :2.29   Median : 0.0000   Median :1081
                                    Mean   :-0.197   Mean   :  Inf   Mean   : 0.04375   Mean   :2.32   Mean   :-0.0838   Mean   :1087
                                    3rd Qu.: 0.994   3rd Qu.:0.402   3rd Qu.: 0.12741   3rd Qu.:2.33   3rd Qu.: 0.0000   3rd Qu.:1085
                                    Max.   : 1.002   Max.   :  Inf   Max.   : 0.14002   Max.   :2.62   Max.   : 0.0000   Max.   :1141
                                    NA's   :1
      llh          converged          n.iter
 Min.   :-568   Mode:logical   Min.   :  0
 1st Qu.:-539   TRUE:11        1st Qu.:123
 Median :-536                  Median :161
 Mean   :-540                  Mean   :191
 3rd Qu.:-536                  3rd Qu.:231
 Max.   :-535                  Max.   :462
```

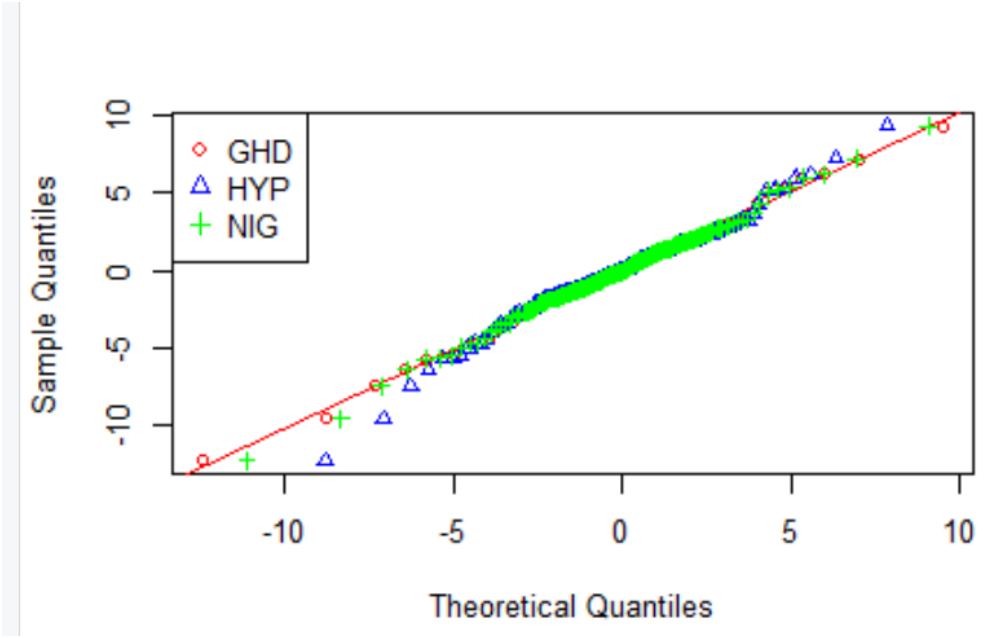
LIK Ratio Test;

```
> LRghdnig <- lik.ratio.test(ghdfit, nigfit)
>                         LRghdnig
$statistic
      L
0.70268

$p.value
[1] 0.40087

$df
[1] 1

$H0
[1] TRUE

>
>
>                         LRghdhyp <- lik.ratio.test(ghdfit, hypfit)
>                         LRghdhyp
$statistic
       L
0.037081

$p.value
[1] 0.010259

$df
[1] 1

$H0
[1] FALSE
```

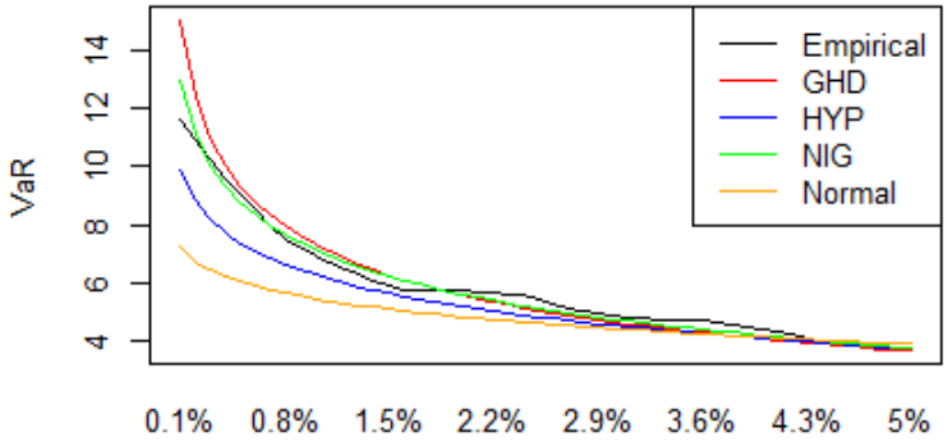
Cisco Density Plot:



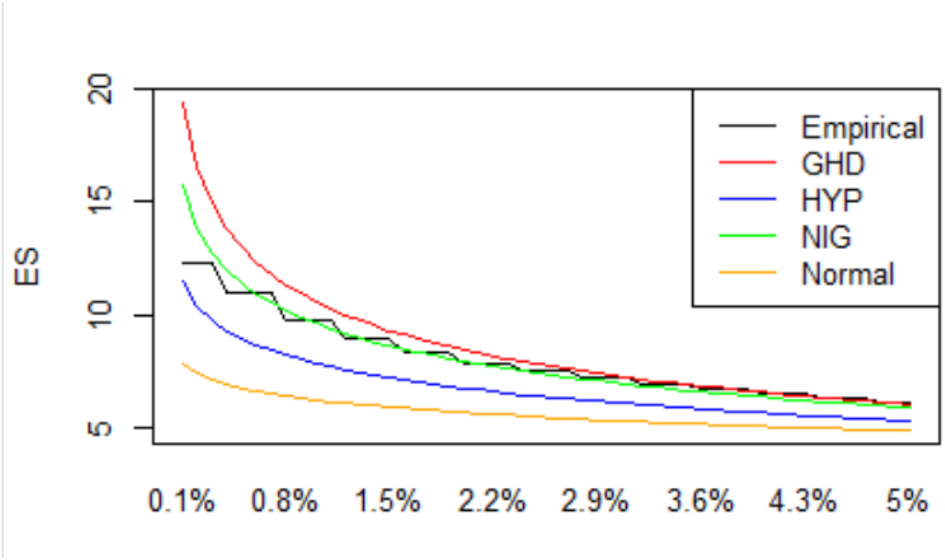
QQ:



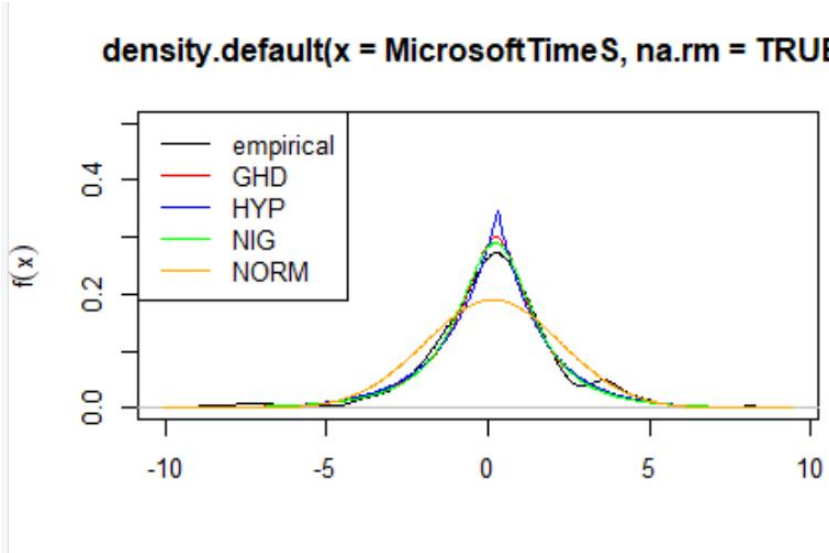
CSCO VAR:

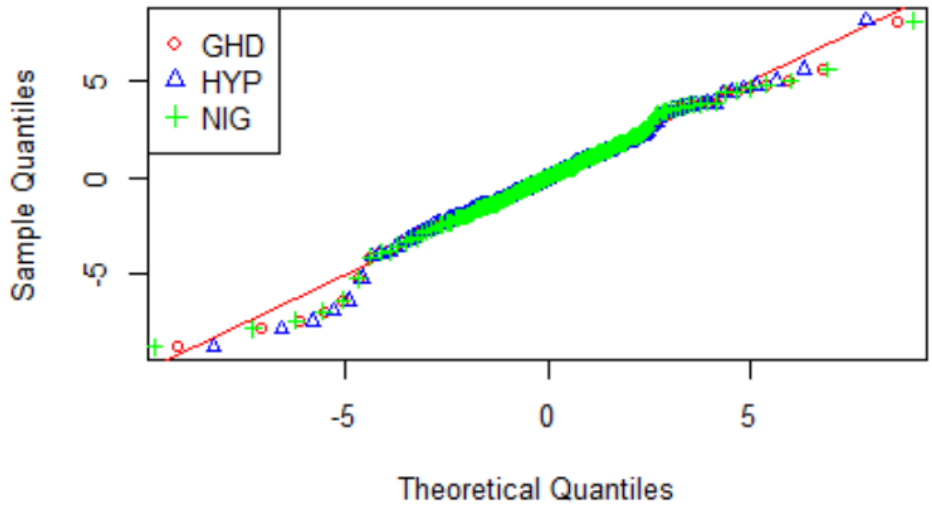


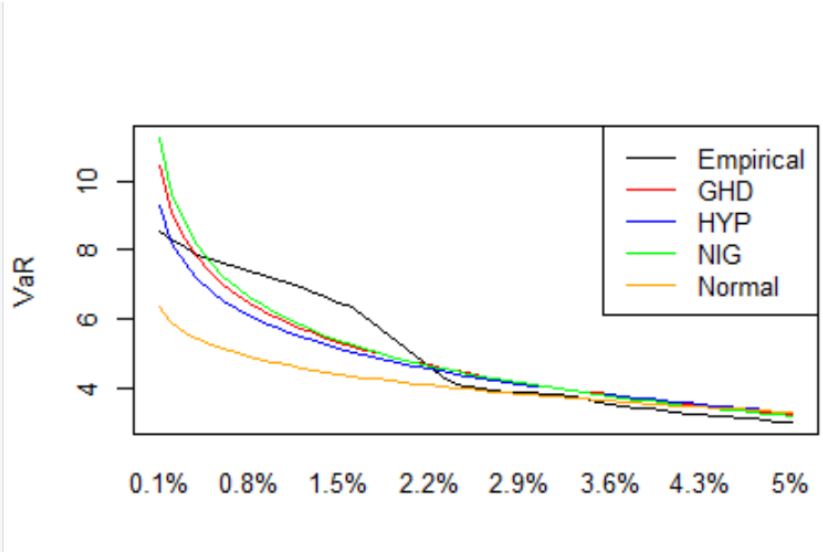
CSCO ES:

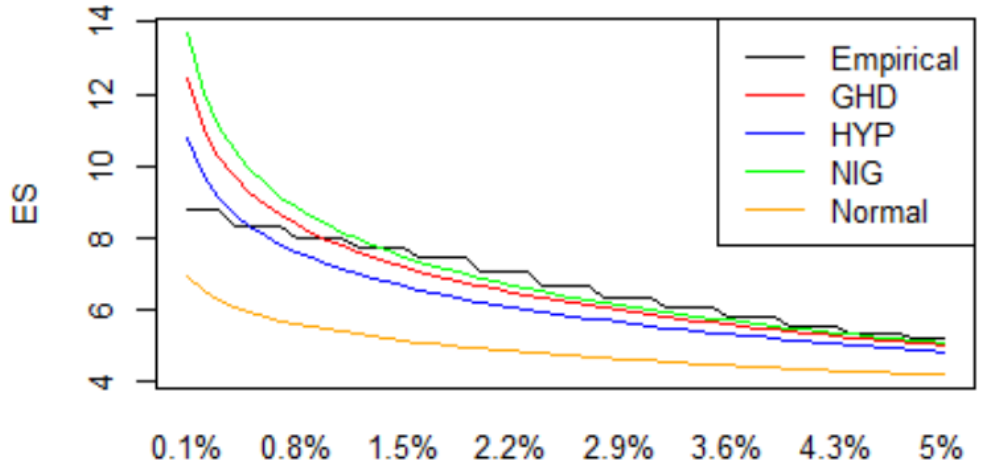

ES
20
15
10
5
0.1%
0.8%
1.5%
2.2%
2.9%
3.6%
4.3%
5%
Empirical
GHD
HYP
NIG
Normal

Microsoft:


density.default(x = MicrosoftTimeS, na.rm = TRUE)
f(x)
0.0
0.2
0.4
-10
-5
0
5
10
empirical
GHD
HYP
NIG
NORM


GHD
HYP
NIG
Sample Quantiles
-5
0
5
-5
0
5
Theoretical Quantiles


Empirical
GHD
HYP
NIG
Normal
VaR
4
6
8
10
0.1%
0.8%
1.5%
2.2%
2.9%
3.6%
4.3%
5%



MSFT Lik ratio test:

```
>                       LRghdhyp <- lik.ratio.test(ghdfit, hypfit)
>                       LRghdnig
$statistic
      L
0.86164

$p.value
[1] 0.58525

$df
[1] 1

$H0
[1] TRUE

>                       LRghdhyp
$statistic
      L
0.76553

$p.value
[1] 0.46478

$df
[1] 1

$H0
[1] TRUE
```

AIC:

```
>                   summary(AIC$fit.table)
   model          symmetric          lambda          alpha.bar           mu              sigma           gamma              aic
Length:11          Mode :logical   Min.   :-1.4606   Min.   :0.0000   Min.   :0.148   Min.   :2.05   Min.   :-0.1561   Min.   :1047
Class :character   FALSE:5         1st Qu.:-0.5000   1st Qu.:0.0000   1st Qu.:0.198   1st Qu.:2.07   1st Qu.:-0.1143   1st Qu.:1048
Mode  :character   TRUE :6         Median : 0.0541   Median :0.0897   Median :0.246   Median :2.10   Median : 0.0000   Median :1049
                                   Mean   : 0.0101   Mean   :   Inf   Mean   :0.234   Mean   :2.14   Mean   :-0.0586   Mean   :1053
                                   3rd Qu.: 0.9573   3rd Qu.:0.4885   3rd Qu.:0.261   3rd Qu.:2.12   3rd Qu.: 0.0000   3rd Qu.:1050
                                   Max.   : 1.0000   Max.   :   Inf   Max.   :0.299   Max.   :2.39   Max.   : 0.0000   Max.   :1092
                                   NA's   :1
      llh          converged          n.iter
Min.   :-544   Mode:logical   Min.   :  0
1st Qu.:-521   TRUE:11        1st Qu.:104
Median :-521                  Median :148
Mean   :-523                  Mean   :168
3rd Qu.:-520                  3rd Qu.:215
Max.   :-520                  Max.   :432
```

5 - Chevron Corp

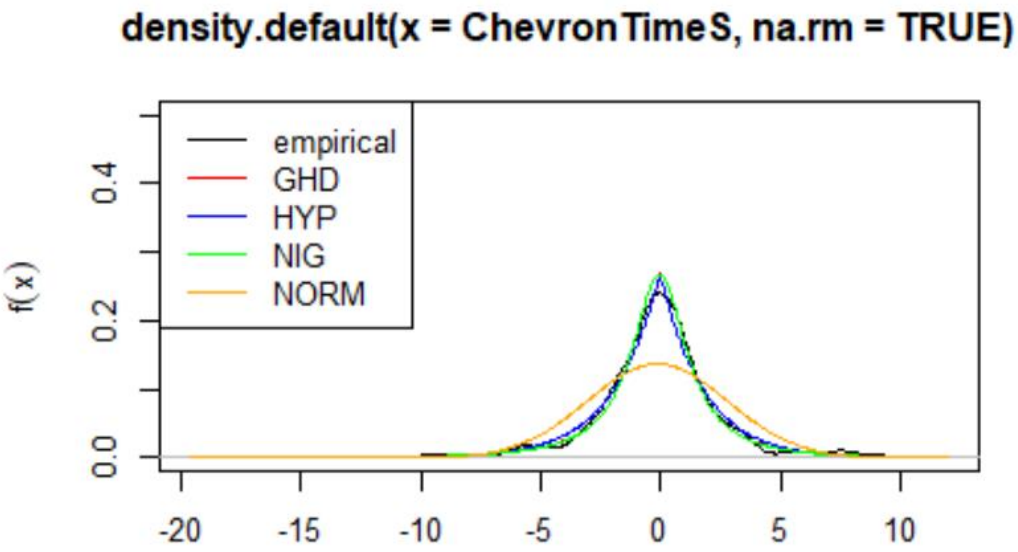
AIC:

```
> summary(AIC$fit.table)
    model           symmetric          lambda          alpha.bar           mu               sigma              gamma              aic
 Length:11          Mode :logical   Min.   :-1.0287   Min.   :0.0000   Min.   :-0.1442   Min.   :   2.59   Min.   :-0.1245   Min.   :1154
 Class :character   FALSE:5         1st Qu.:-0.5000   1st Qu.:0.0000   1st Qu.:-0.0813   1st Qu.:   2.69   1st Qu.:-0.0678   1st Qu.:1156
 Mode  :character   TRUE :6         Median :-0.4741   Median :0.0466   Median :-0.0785   Median :   2.93   Median : 0.0000   Median :1159
                                    Mean   :-0.0475   Mean   :   Inf   Mean   :-0.0712   Mean   : 190.06   Mean   : 0.1100   Mean   :1169
                                    3rd Qu.: 0.7544   3rd Qu.:0.2341   3rd Qu.:-0.0533   3rd Qu.:   2.94   3rd Qu.: 0.0000   3rd Qu.:1166
                                    Max.   : 1.0000   Max.   :   Inf   Max.   :-0.0198   Max.   :2057.04   Max.   : 1.5822   Max.   :1257
                                    NA's   :1
      llh          converged           n.iter
 Min.   :-627   Mode:logical   Min.   :  0
 1st Qu.:-579   TRUE:11        1st Qu.:146
 Median :-575                  Median :164
 Mean   :-581                  Mean   :186
 3rd Qu.:-574                  3rd Qu.:241
 Max.   :-574                  Max.   :334
```

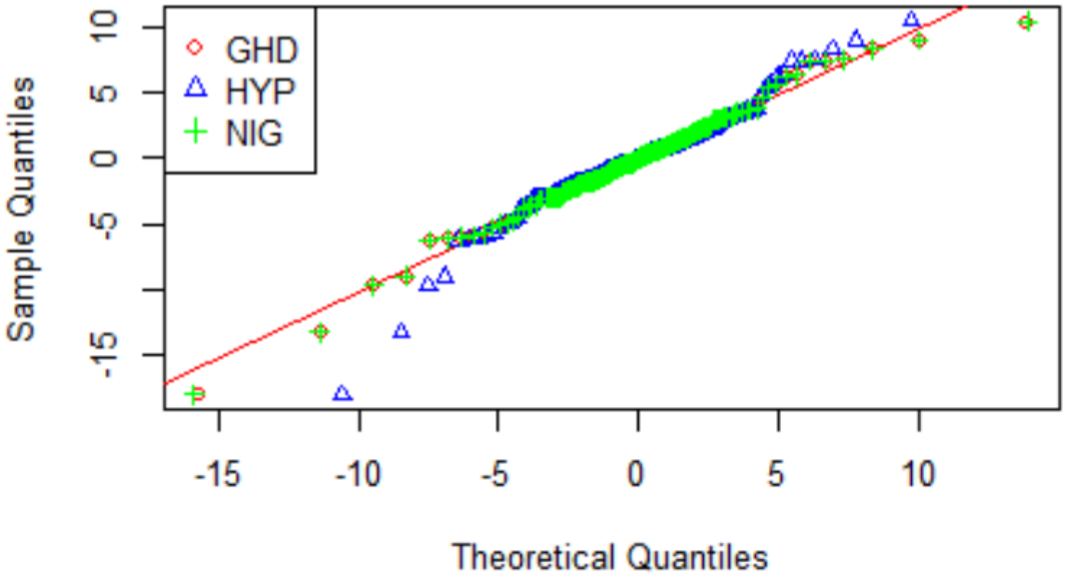
LIK Ratio Test:

```
> LRghdnig
$statistic
      L
0.99778

$p.value
[1] 0.94687

$df
[1] 1

$H0
[1] TRUE

> LRghdhyp
$statistic
       L
0.0025727

$p.value
[1] 0.00055367

$df
[1] 1

$H0
[1] FALSE
```

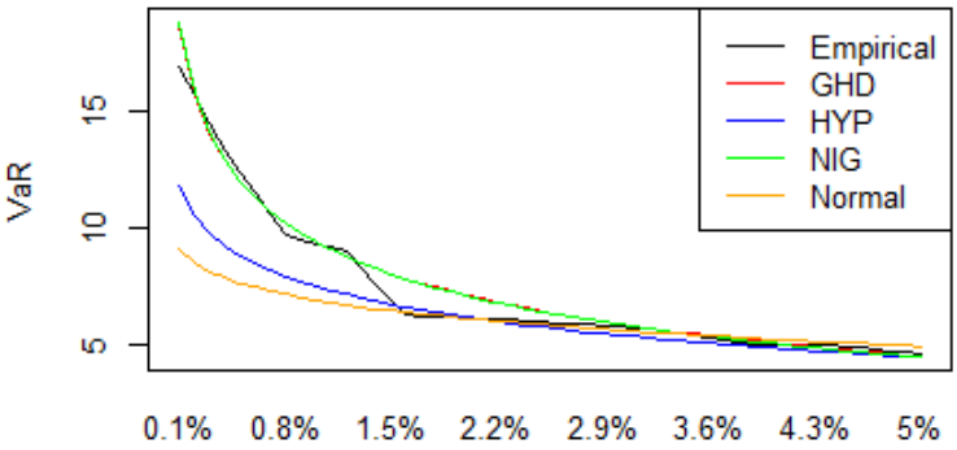
Density:



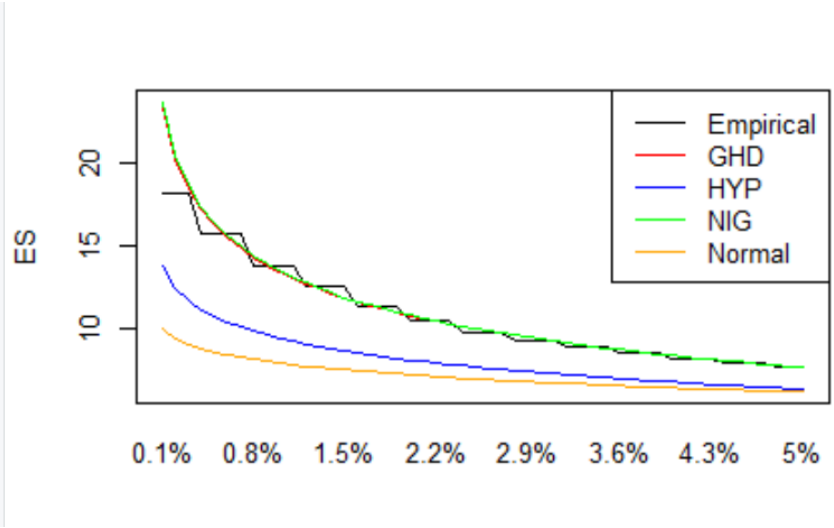
QQ Plot:


GHD
HYP
NIG
Sample Quantiles
Theoretical Quantiles
-15
-10
-5
0
5
10

VAR:


Empirical
GHD
HYP
NIG
Normal
VaR
5
10
15
0.1%
0.8%
1.5%
2.2%
2.9%
3.6%
4.3%
5%

ES:



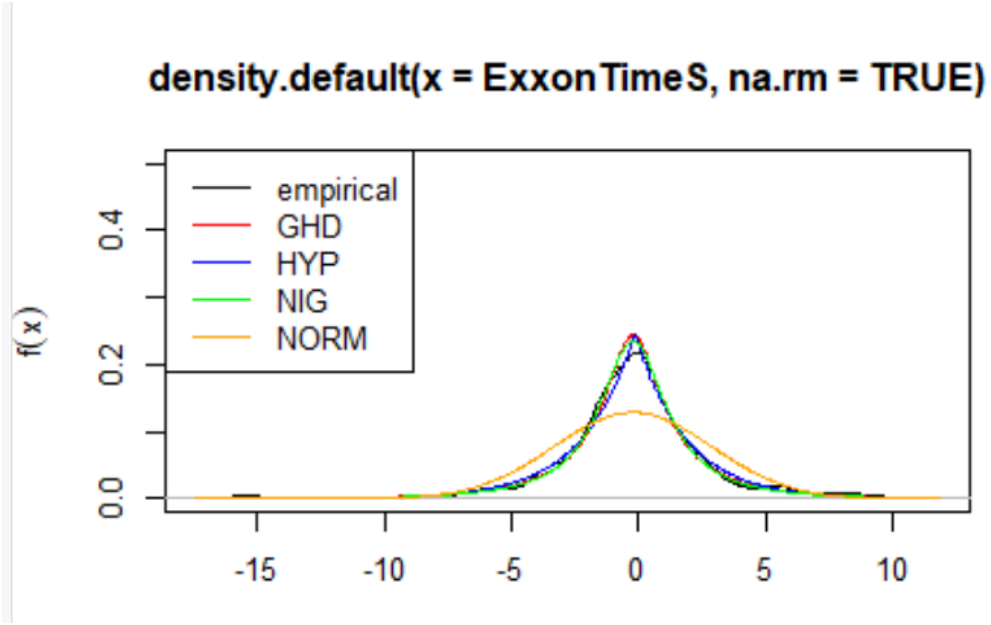
6 - Exxon Mobil:

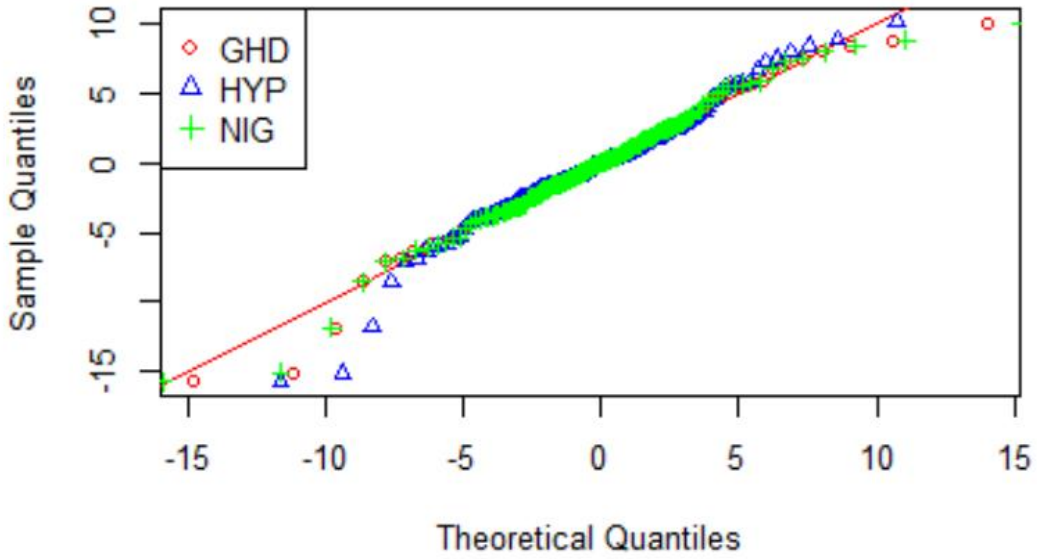
AIC:

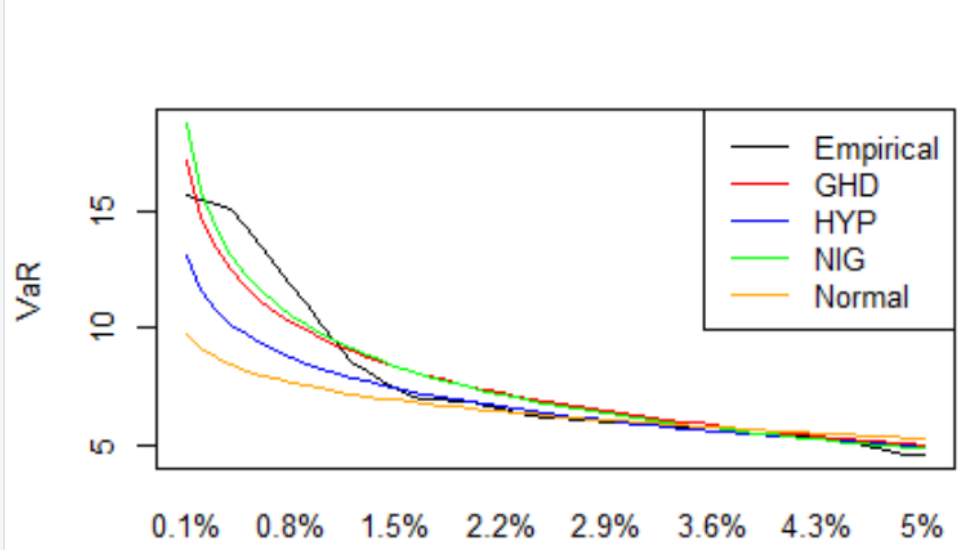
```
> summary(AIC$fit.table)
    model           symmetric          lambda          alpha.bar            mu                sigma          gamma              aic
 Length:11          Mode :logical   Min.   :-1.0807   Min.   :0.000000   Min.   :-0.2228   Min.   :2.86   Min.   :-0.1241   Min.   :1209
 Class :character   FALSE:5         1st Qu.:-0.5000   1st Qu.:0.000000   1st Qu.:-0.2144   1st Qu.:3.00   1st Qu.:-0.0255   1st Qu.:1211
 Mode  :character   TRUE :6         Median :-0.0919   Median :0.000031   Median :-0.2059   Median :3.11   Median : 0.0000   Median :1212
                                    Mean   : 0.0118   Mean   :     Inf   Mean   :-0.1799   Mean   :3.56   Mean   :-0.0264   Mean   :1219
                                    3rd Qu.: 0.7394   3rd Qu.:0.274178   3rd Qu.:-0.1570   3rd Qu.:3.17   3rd Qu.: 0.0000   3rd Qu.:1214
                                    Max.   : 1.0000   Max.   :     Inf   Max.   :-0.0665   Max.   :5.90   Max.   : 0.0000   Max.   :1287
                                    NA's   :1
      llh          converged          n.iter
 Min.   :-642   Mode:logical   Min.   :  0
 1st Qu.:-604   TRUE:11        1st Qu.:141
 Median :-602                  Median :191
 Mean   :-606                  Mean   :223
 3rd Qu.:-601                  3rd Qu.:237
 Max.   :-601                  Max.   :724
```

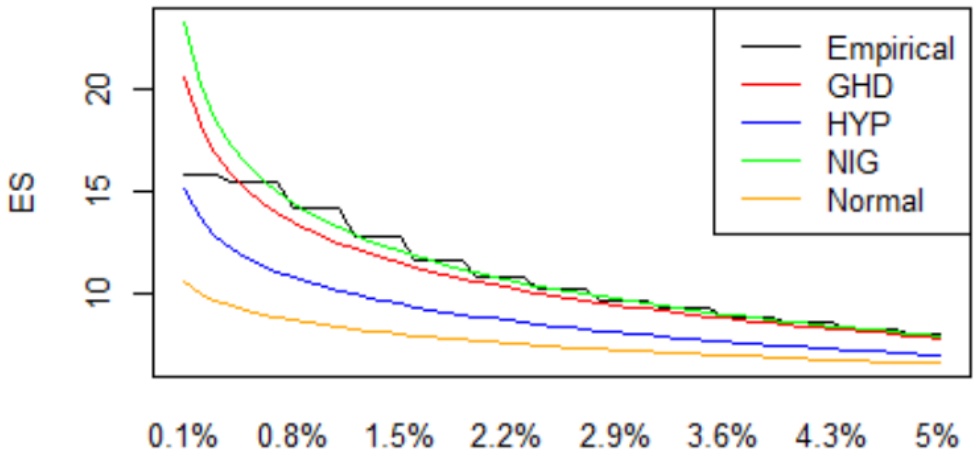
LIK Ratio Test:

```
> LRghdnig
$statistic
      L
0.74505

$p.value
[1] 0.44296

$df
[1] 1

$HO
[1] TRUE

> LRghdhyp
$statistic
       L
0.056503

$p.value
[1] 0.016518

$df
[1] 1

$HO
[1] FALSE
```


density.default(x = ExxonTimeS, na.rm = TRUE)
empirical
GHD
HYP
NIG
NORM
f(x)
0.0
0.2
0.4
-15
-10
-5
0
5
10


GHD
HYP
NIG
Sample Quantiles
-15
-5
0
5
10
-15
-10
-5
0
5
10
15
Theoretical Quantiles





Calculate and plot the Value at Risk VaR (using all models)

Model Description:

To find the VaR of significance level alpha, we used the quantile command. The first argument is the negative of the simulated portfolio P&L and the second argument is the 1-alpha confidence level.

We plotted the VaR by proving the ranges for hyp.VaR, nig.VaR, ghd.VaR, nor.VaR, and emp.VaR.

To obtain the density of the simulated portfolio P&L, we used the density command. Then we added two vertical lines using the abline command (estimates of the VaR) and created a sequence of 1000 numbers between the smallest and the largest simulated P&L. This created bounds for the normal density plot.

We used the dnorm command to provide the values of the probability density function for the normal distribution. (dnorm: x vector created above, mean of the simulated portfolio P&L, sd of the simulated portfolio P&L). Then used the lines command to add the normal density plot as a dashed bell curve onto the plot.

Calculate and plot the ES (using all models)

ES or Estimated Shortfall is also known as tail VaR and tail loss. We calculated Historic ES by taking the average portfolio loss that fails short of the Historical VaR estimate.

To calculate ES, we found the average of the losses based on a threshold level that takes into consideration the 1% 1-Day Historical ES and for 5% 1-Day Historical ES.

Then we extracted portfolio losses more than VaR and computed average losses more than VaR. We then took the average of the returns in each of these data objects.
ES is expected to be larger than VaR. We used a combination of rbind and cbind to bring together all the resulting calculations of Historical and Gaussian VaR and ES.

We plotted the ES using the ranges hyp.ES, nig.ES, ghd.ES, nor.ES, emp.ES

Portfolio Optimization to determine optimized portfolio weights:

Objective:

We have explored both the Markwitz technique and the Quadratic Programming approach. Our objective is to find the combination of securities that yield the minimum risk for a specific level of return.

Step 1: Finding the monthly return and binding the two stocks - AMGN and JNJ together.

```
> Ret.monthly <- cbind(AMGN.ret[-1,],JNJ.ret[-1])
> Ret.monthly[c(1:3,nrow(Ret.monthly)),]
         AMGN.Ret JNJ.Ret
Aug 2010  -0.0790  0.1221
Sep 2010  -0.0553 -0.1179
Oct 2010  -0.4602 -0.2752
Jun 2020  -0.3910 -0.0902
> AMGN.Avg <- mean(Ret.monthly$AMGN.Ret)
>     AMGN.Avg
[1] 0.137
>     AMGN.sd <- sd(Ret.monthly$AMGN.Ret)
>     AMGN.sd
[1] 0.674
>     JNJ.Avg <- mean(Ret.monthly$JNJ.Ret)
>     JNJ.Avg
[1] 0.0997
>     JNJ.sd <- sd(Ret.monthly$JNJ.Ret)
>     JNJ.sd
[1] 0.56
```

Variance-Covariance Matrix and return portfolio vector:

Model Description:

We imported the returns data and converted it into a matrix. Then we calculated the variance-covariance matrix of returns. Then we constructed the target portfolio return vector.

```
> covar
           JNJ.Ret
AMGN.Ret     0.216
> options(scipen = 100)
> covar <- covar[1,1]
> covar
[1] 0.216
```

We created a series of stock weights: To find all the possible combinations of AMGN and JNJ stock, we created a series with all possible weight combinations.

```
> Portfolio <- data.frame(seq(0,1,by=.01))
>     names(Portfolio) <- paste("AMGN.wgt")
>     Portfolio[1:5,]
[1] 0.00 0.01 0.02 0.03 0.04
>     Portfolio[(nrow(Portfolio)-5):nrow(Portfolio),]
[1] 0.95 0.96 0.97 0.98 0.99 1.00
```

Output:

```
> Portfolio[c(1:3,nrow(Portfolio)),]
    AMGN.wgt JNJ.wgt
1       0.00    1.00
2       0.01    0.99
3       0.02    0.98
101     1.00    0.00
```

Calculating a probability for each combination of stocks:

```
>     Portfolio[c(1:3,nrow(Portfolio)),]
    AMGN.wgt JNJ.wgt PortRet PortRisk
1       0.00    1.00  0.0997    0.560
2       0.01    0.99  0.1001    0.558
3       0.02    0.98  0.1005    0.556
101     1.00    0.00  0.1373    0.674
```

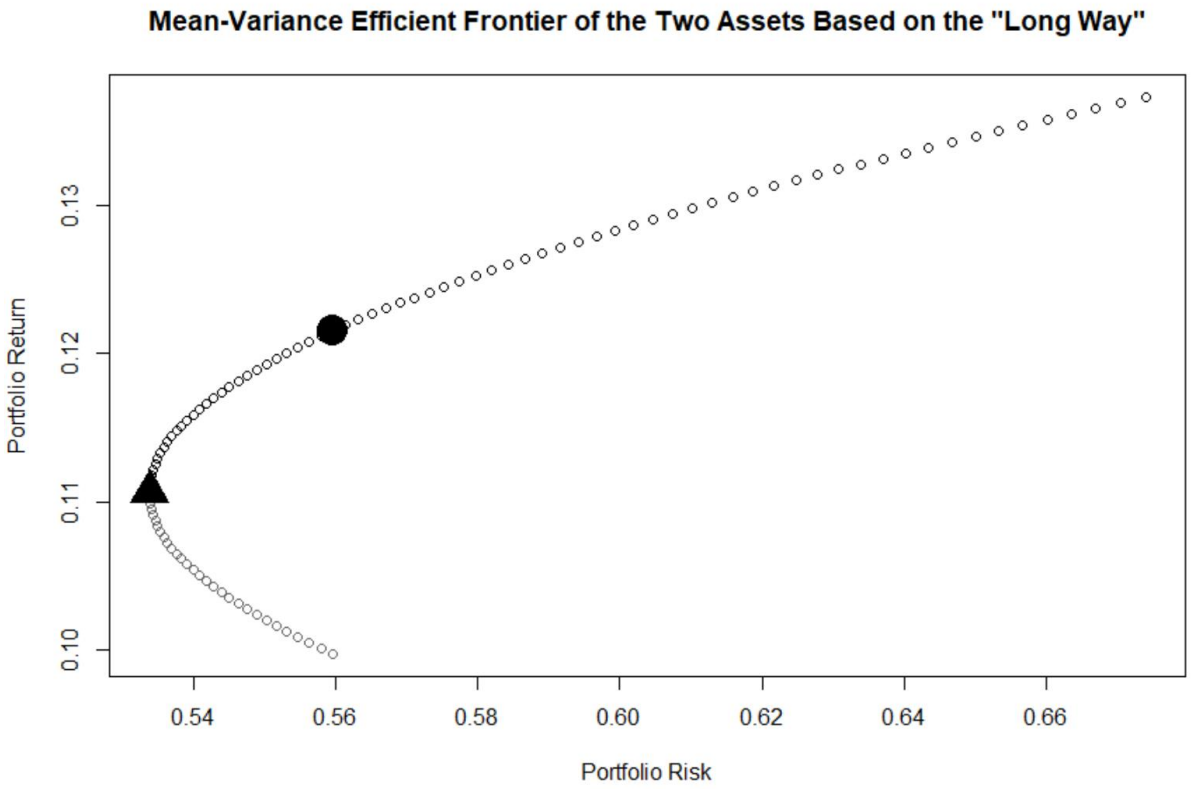
Identification of tangency portfolio:

```
>     tangency.port
   AMGN.wgt JNJ.wgt PortRet PortRisk Sharpe
59     0.58    0.42   0.122     0.56  0.217
```

Building the efficient frontier:

```
>     eff.frontier[c(1:3,nrow(eff.frontier)),]
    AMGN.wgt JNJ.wgt PortRet PortRisk Sharpe
30      0.29    0.71   0.111    0.534  0.207
31      0.30    0.70   0.111    0.534  0.208
32      0.31    0.69   0.111    0.534  0.209
101     1.00    0.00   0.137    0.674  0.204
```

Plotting the MV Efficient Frontier of the two assets based on 'Long way' method:



Running the optimizer:

Model Description

For the other 4 stocks, in our portfolio, we used the Optimizer Quadprog approach. We created a series of dummy portfolio weight vectors and then ran the quadprog optimizer. Then we combined Portfolio Returns, calculated the Portfolio Standard Deviations, and Portfolio Weights. Next we identified the Minimum Variance Portfolio and the Tangency portfolio. Finally, we determined the most efficient portfolio and plotted the efficient frontier.

Calculation of monthly returns:

```
> Ret.monthly[c(1:3,nrow(Ret.monthly)),]
          CSCO.Ret MSFT.Ret CVX.Ret XOM.Ret
Aug 2010    1.239  -0.2090 -0.4558 -0.0807
Sep 2010   -0.398  -0.0728  0.0968 -0.0565
Oct 2010   -0.216   0.8640  0.2315 -0.2433
Jun 2020   -0.498  -0.1859 -0.5144 -0.1442
>
```

```
> head(mat.ret)
       CSCO    MSFT     CVX     XOM
[1,]  1.239 -0.2090 -0.4558 -0.0807
[2,] -0.398 -0.0728  0.0968 -0.0565
[3,] -0.216  0.8640  0.2315 -0.2433
[4,]  0.422 -0.3407 -0.1683  0.4287
[5,] -0.548 -0.6712 -0.5029 -0.4451
[6,]  0.408  1.6272  1.1673  1.2597
>
```

Calculation of Variance - Covariance matrix:

```
> VCOV <- cov(mat.ret)
> VCOV
      CSCO  MSFT   CVX   XOM
CSCO 0.423 0.222 0.123 0.118
MSFT 0.222 0.594 0.316 0.188
CVX  0.123 0.316 0.437 0.225
XOM  0.118 0.188 0.225 0.256
```

Running the Quadprog optimizer:

```
> library(quadprog)
> for (i in 1:increments) {
+       Dmat <- 2*VCOV
+       dvec <- c(rep(0,length(avg.ret)))
+       Amat <- cbind(rep(1,length(avg.ret)),avg.ret,
+                     diag(1,nrow=ncol(Ret.monthly)))
+       bvec <- c(1,tgt.ret[i],rep(0,ncol(Ret.monthly)))
+       soln <- solve.QP(Dmat,dvec,Amat,bvec=bvec,meq=2)
+       tgt.sd[i] <- sqrt(soln$value)
+       wgt[i,] <- soln$solution}
> head(tgt.sd)
[1] 0.506 0.501 0.497 0.493 0.489 0.485
> tail(tgt.sd)
[1] 0.714 0.725 0.736 0.747 0.759 0.770
> options(scipen=100)
> head(wgt)
                      [,1]                    [,2] [,3]  [,4]
[1,] 0.00000000000000025 -0.000000000000000139    0 1.000
[2,] 0.01708757983523534  0.000000000000000139    0 0.983
[3,] 0.03417515967047000  0.000000000000000000    0 0.966
[4,] 0.05126273950570509  0.000000000000000139    0 0.949
[5,] 0.06835031934093987  0.000000000000000000    0 0.932
[6,] 0.08543789917617499  0.000000000000000000    0 0.915
>
```

Pasting names of the asset to the weights:

```
> head(wgt)
     wgt.CSCO                 wgt.MSFT wgt.CVX wgt.XOM
[1,]   0.0000 -0.0000000000000000139       0   1.000
[2,]   0.0171  0.0000000000000000139       0   0.983
[3,]   0.0342  0.0000000000000000000       0   0.966
[4,]   0.0513  0.0000000000000000139       0   0.949
[5,]   0.0684  0.0000000000000000000       0   0.932
[6,]   0.0854  0.0000000000000000000       0   0.915
```

Identify the minimum variance portfolio. This indicates the smallest risk:

```
> minvar.port <- subset(tgt.port,tgt.port$tgt.sd==min(tgt.port$tgt.sd))
> minvar.port
   tgt.ret tgt.sd wgt.CSCO wgt.MSFT wgt.CVX wgt.XOM
24   0.107   0.46    0.302  0.00748    0.08    0.61
```

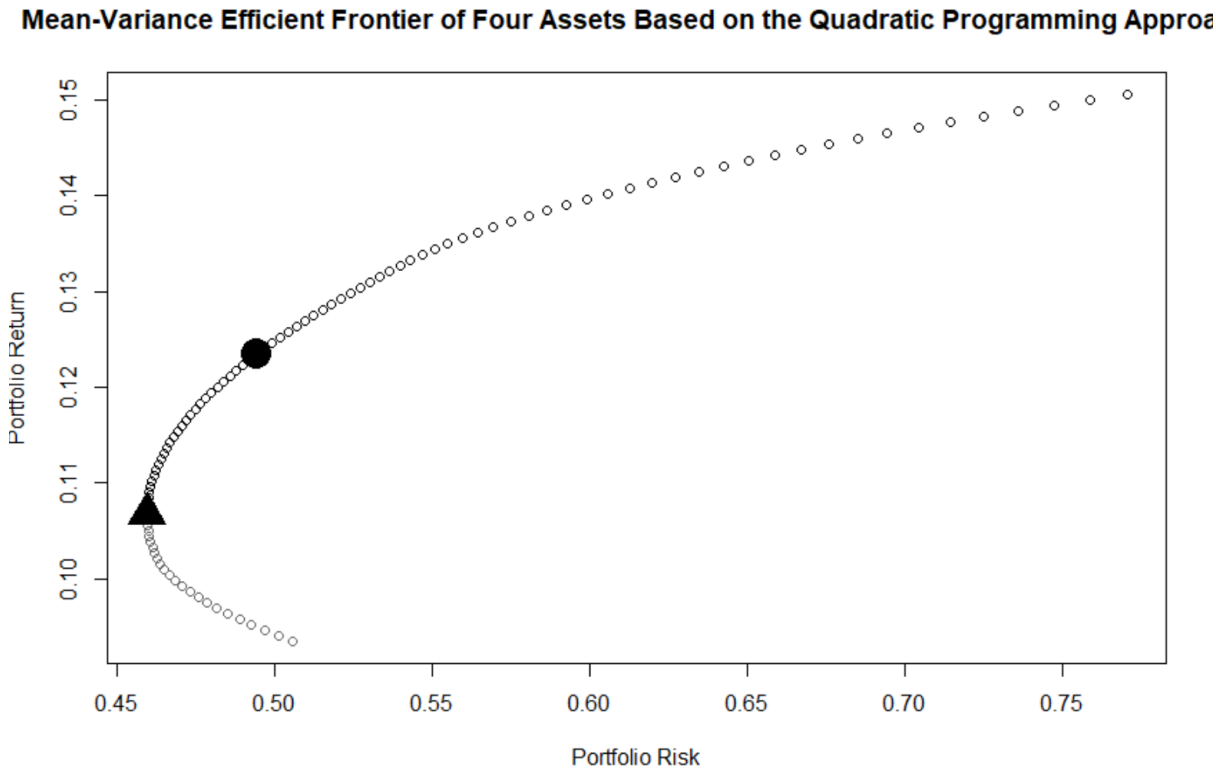
Identify the tangency portfolio:

```
> head(tgt.port)
  tgt.ret tgt.sd wgt.CSCO                wgt.MSFT wgt.CVX wgt.XOM Sharpe
1  0.0935  0.506   0.0000 -0.0000000000000000139       0   1.000  0.185
2  0.0941  0.501   0.0171  0.0000000000000000139       0   0.983  0.188
3  0.0947  0.497   0.0342  0.0000000000000000000       0   0.966  0.190
4  0.0953  0.493   0.0513  0.0000000000000000139       0   0.949  0.193
5  0.0958  0.489   0.0684  0.0000000000000000000       0   0.932  0.196
6  0.0964  0.485   0.0854  0.0000000000000000000       0   0.915  0.199
> tangency.port <- subset(tgt.port,tgt.port$Sharpe==max(tgt.port$Sharpe))
> tangency.port
   tgt.ret tgt.sd wgt.CSCO wgt.MSFT wgt.CVX wgt.XOM Sharpe
53   0.123  0.495    0.379    0.168   0.231   0.222   0.25
```

Building the efficient frontier:

```
> eff.frontier <- subset(tgt.port,tgt.port$tgt.ret>=minvar.port$tgt.ret)
> eff.frontier[c(1:3,nrow(eff.frontier)),]
    tgt.ret tgt.sd              wgt.CSCO wgt.MSFT wgt.CVX              wgt.XOM Sharpe
24    0.107   0.46 0.3023821754616868307  0.00748  0.0800 0.6101450855040859711  0.232
25    0.107   0.46 0.3050201720936994709  0.01302  0.0852 0.5967518807619440757  0.233
26    0.108   0.46 0.3076581687257122222  0.01855  0.0904 0.5833586760198018473  0.235
100   0.151   0.77 0.0000000000000000278  0.00000  0.0000 0.0000000000000000209  0.195
```

Plotting the Efficient portfolio based on Quadratic programming approach:



Conclusion/recommendation:

The circle represents the tangency portfolio and the Triangle represents the minimum variance portfolio. On the x-axis, we plotted the Portfolio risk and the estimated Portfolio return on the Y axis. The returns are higher with more risk. The best trade-off between risk and gain is represented by the shapes above. For the minimum variance portfolio, at 0.45 portfolio risk gives a return of 0.108. For the tangency portfolio, at 0.5, portfolio risk gives a return of 0.125.

Appendix:

R-Code.

ANLY 515 Project Final R Code.docx

R Studio File:

Final ANLY Project 515 August 4 2 PM.R